Improving Language Understanding by Generative Pre-Training

Introduction

대부분의 딥러닝 방법론들은 상당한 양의 labeled data를 필요로 하는데, 이는 labeled data의 부족으로 인해 해당 방법론이 많은 domain에 적용되지 못하는 이유가 되곤 한다.

이러한 상황에서, unlabeled data로부터 linguistic information을 활용할 수 있는 model은 비용과 시간이 많이 필요한 data labling을 대체할 수 있는 좋은 수단이다.

또한, labeled data가 충분하여 considerable supervision이 가능하더라도 unsupervised fashion으로부터 좋은 representation을 학습하는 것은 성능의 향상을 불러일으킬 수 있다. (논문에서는 이에 대한 근거로 pretrained word embedding을 사용하여 성능 향상을 일으킨 선행 연구들을 제시한다.)

그러나, 이렇게 unlabeled data로부터 linguistic information을 활용하는 것은 다음과 같은 2가지의 해결 과제가 있다.

다른 model에 전이(transfer)하여 사용하기 좋은 형태의 text representation을 학습하기 위해 model을 어떤 목적으로 최적화해야 하는지 분명하지 않음
학습된 representation을 target task에 효과적으로 전이(transfer)하는 방법에 대한 합의가 없다.
논문이 작성될 때 기준으로는, 이러한 문제를 해결하기 위해 복잡한 learning schemes를 이용하여 model의 architecture를 task-specific 하게 변경하고, 보조 목적 함수를 추가하는 방법을 조합했었다.

이러한 불확실성은 자연어 처리에서 효과적인 준지도 학습(semi-supervised learning) 접근 방법을 발전시키는 것을 어렵게 만들었다.

저자들은 이러한 문제를 해결하고자 unsupervised pre-training과 supervised fine-tuning을 조합한 새로운 semi-supervised의 접근법을 제시한다.

이 연구의 목적은 universal representation을 다양한 범위의 task(text classification, question answering... 등등)에 transfer 시키는 것이다. 이때, unlabled corpus와 target task가 같은 domain 일 필요도 없다.

논문에서는 다음과 같은 두 단계의 training 과정을 제시한다.

Neural network model의 초기 파라미터(initial parameters)를 학습하기 위해 unlabeled data에 language modeling을 적용한다.
학습된 initial parameters를 target task에 적용시켜 supevised learning을 진행한다.
이러한 과정들에는 transformer architecture가 사용된다. 또한, transfer 과정에서는 traversal-style approach로부터 파생된 task-specific input adaptation을 사용하여 text input이 하나의 연속된 token sequence가 될 수 있게끔 한다. 이러한 input adaptation은 pre-trained model의 architecture를 최소한의 수정으로도 효과적으로 fine-tuning 할 수 있게끔 한다.

Related Work

논문에서는 related work를 다음과 같은 3개의 부분으로 나눠서 설명한다.

- Semi-supervised learning for NLP (NLP에서의 준지도 학습)
- Unsupervised pre-training (비지도 학습)
- Auxiliary training objective (보조 목적 함수)

하나씩 차근차근 살펴보자

Semi-supervised learning for NLP

NLP에서의 semi-supervised learning은 sequence labeling과 text classification에 적용되어 큰 주목을 받아왔다.

초기의 접근 방법은 unlabeled data를 word-level(단어 수준)이나 phrase-level(구문 수준)에서의 통계치를 계산하는 데 사용하고, 이 계산된 통계치를 supervised model의 feature로 사용하는 것이었다. 또한, 현재 소개하고 있는 연구가 진행될 시점의 이전 몇 년간, 연구자들은 unlabeled data로부터 학습된 word embedding이 다양한 task에서 성능 향상을 이끌어냄을 증명해왔다.

그러나, 이러한 연구들은 대부분 word-level(단어 수준)의 정보를 transfer 했지만 더 높은 수준에서의 정보를 담아내지는 못했다.

최근 연구들은(2018년 기준 최근이다.) unlabeled corpus로부터 학습하고, 다양한 task에 대해 text를 적합한 representataion으로 encode 할 수 있는 phrase-level(구문 수준)이나 sentence-level(문장 수준)의 embedding을 연구해오고 있다.

Unsupervised pre-training

Unsupervised pre-training은 semi-supervised learning의 특별한 경우이며, supervised learning의 objective를 변경하는 것 대신 좋은 initialization point를 찾는 것이 목적이다. 이에 관해 image classification과 regression task에서 연구들이 진행되었다.

또한, pre-training이 regularization scheme처럼 작동하여 deep neural network의 generalization을 용이하게 한다는 연구도 제안되었으며, 최근에는 이러한 방법론이 image classification에 국한되지 않고 speech recognition, entity disambiguation, machine translation과 같은 다양한 영역에서 deep neural network의 학습을 돕기 위한 방법론으로 채택되고 있다.

NLP에서도 text classification에 unsupervised pre-training을 활용한 연구가 진행되었다. 그런데, 해당 연구에서는 pre-train이 linguistic information을 포착하는데 도움을 주었지만 LSTM architecture의 한계로 인해 model의 예측 능력이 short range로 국한되었다.

본 연구에서는 더 넓은 범위의 언어적 구조를 학습하기 위해 transformer network를 사용한다.

Auxiliary training objective

Auxiliary objective(보조 목적 함수)를 unsupervised learning에 추가하는 것은 semi-supervised learning의 대안적인 형태이다.

NLP task에 보조 목적 함수를 사용하여 성능을 향상한 선행 연구가 존재하고, 본 연구에서도 보조 목적 함수를 추가하여 학습을 진행하였지만, unsupervised pre-training로도 target task에 대한 linguistic information을 충분히 학습했음을 보인다.

Framework

앞서 잠깐 언급한 것처럼, 본 연구에서 training procedure는 두 단계로 나뉜다.

1. Large corpus로 high-capacity language model을 학습한다. (Unsupervised pre-training)
2. 각 task에 대해 model을 adapt 하는 fine-tuning을 한다.(Supervised fine-tuning)

각각의 단계를 차례대로 살펴보겠다.

Unsupervised pre-training

unsupervised corpus of token $U\{u_1, \ldots, u_n\}$이 주어진다고 해보자. 이를 log-likelihood를 최대로 하는 standard language modeling objective에 적용하면 다음과 같은 수식이 탄생한다.

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

이 수식에서 k는 context window의 size이며, 조건부 확률 P는 parameter Θ를 가진 neural network에 의해 구성되며, parameter Θ는 stochastic gradient descent로 train 된다.

이러한 language modeling에 관해서는 아래의 포스팅에서 다루었으니 참고하면 좋을 것 같다.

언어 모델(Language Model) 이란?

본 연구에서는 language modeling을 위해 tranformer의 변형인 multi-layer transformer decoder를 사용하였다.

transformer에 관련한 자세한 내용도 포스팅하였는데, 아래의 링크를 참고 바란다.

[논문 리뷰] Attention is all you need - transformer란?

이러한 transformer는 multi-headed self-attention을 input context vector에 적용하고, position-wise feed forward network에 통과시켜 target token의 분포를 output으로 내놓는다.

즉, 아래와 같은 과정을 거친다.

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

위 수식에서 $U = (u_{-k}, \ldots, u_{-1})$는 context vector of tokens, n은 layer의 개수, W_e는 embedding matrix를 의미하며 W_p는 position embedding matrix를 나타낸다.

Supervised fine-tuning

Unsupervised pre-training을 마친 이후, supervised target task로 parameter를 adapt 시키는 fine-tuning을 진행한다.

각각의 instance가 input token으로 이루어진 sequence, $x^1, \ldots, x^m$과 label y로 이루어져 있는 dataset C가 있다고 가정해보자. 이 input dataset은 pre-trained model에 입력으로 들어가게 되고, 이를 통해 우리는 마지막 transformer block의 activation인 h_l^m을 얻게 된다. 이 activation h_l^m은 parameter W_y를 가진 linear output layer에 입력되어 최종적으로 y를 predict 하게 된다.

이러한 과정을 수식으로 나타내면 다음과 같다.

$$P(y|x^1, \dots, x^m) = \text{softmax}(h_l^m W_y)$$

또한, 위 과정은 다음의 목적함수를 최대화하는 방향으로 학습된다.

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

(즉, negative-log-likelihood를 최소화하는 방향으로 학습한다.)

그런데, 논문에서는 이러한 fine-tuning의 objective function $L_2(C)$에 auxiliary objective(보조 목적함수)로 language modeling을 추가하고, 이에 대한 weight λ를 두어 새로운 목적함수 $L_3(C)$를 만들고, 이를 optimizing 한다고 한다.

논문에서는 이러한 auxiliary objective(보조 목적함수)를 추가하는 것은 supervised model의 generalization을 향상하고, convergence를 촉진시키는 이점이 있다고 설명한다.

따라서, 다음과 같은 목적 함수를 optimize 하게 된다.

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

(L_1과 L_2는 위의 수식들 참고)

Task specific input transformations

text classification과 같은 task에서는 위에서 언급된 방법대로 곧바로 fine-tuning이 가능하다. 그러나, question answering과 textual entailment와 같이 input이 문장의 쌍으로 이루어져 있거나 triplets을 형성하고 있는, structured input을 가지는 task에서는 그렇지 못하다.

논문에서 제안하는 pre-trained modeld은 contiguous sequence of text(연속된 문장)에서 train 되었기 때문에, 우리는 이러한 task에 model을 적용하기 위해서는 약간의 수정이 필요하다.

기존의 연구들은 transferred representation 위에 특정 task에 맞는 architecture를 학습하는 방법을 제시했지만,

(ELMo의 경우이다. 실제로 논문에서도 이 방법론을 소개하면서 ELMo가 소개된 논문을 제시하였다.)

이러한 방법은 상당한 양의 task-specific customization을 다시 도입하며, 이러한 추가적인 task-specific architecture에는 transfer learning을 사용하지 않는다는 단점이 있다.

대신, 논문에서는 structured input을 pre-trained model이 처리할 수 있는 ordered sequence로 변환해주는 travelsal-style apporoach를 제안한다. 이러한 input transformation을 통해, task에 따라 architecture를 광범위하게 변경하지 않아도 된다는 장점이 있다.

논문에서는 각각의 task에 대해 어떻게 transformation 해야 하는지 제시하는데, 해당 내용은 다음과 같다.

- **Textual entailment** : premise p와 hypothesis h sequence를 concat 하고, 사이에는 delimeter token $를 넣어준다.
- **Similarity** : 유사도를 구할 때에는 두 개의 sentence에 정해진 순서가 따로 없다. 이를 빈영하여, input sequence를 가능한 두 경우 모두의 sentence ordering으로 수정해준다. 예를 들어 text1, text2가 있다고 가정해보자. text 1, text2의 순서도 가능할 것이고 text2, text1의 순서도 가능할 것이다. 이렇게 가능한 순서 모두 text1과 text2를 concat 해주고, 두 문장 사이에 delimeter token $를 넣어준다. 이때, 생성된 두 input sequence에 대한 sequence representation h_l^m 을 구하는

각각의 과정은 **독립적**으로 이루어진다.

- **Question answering and commonsense reasoning** : 해당 task는 context document z, question q, set of possible answer a_k가 주어지는데, 이를 바탕으로 context document z와 question q, 그리고 가능한 모든 answer을 각각 concat 한다.(이 때도 마찬가지로 문장 사이에 delimeter token $를 넣어준다.) 유사도를 구할 때와 마찬가지로, 가능한 모든 answer에 대한 각각의 input sequence를 통해 sequence representation h_l^m 을 구하는 과정은 **독립적**으로 이루어진다.

Experiments

Setup

먼저, Unsupervised pre-training을 위해 7000권이 넘는 다양한 분야의 미출판 서적에 관련된 dataset인 BookCorpus dataset을 사용한다. 이 BookCorpus dataset은 긴 길이의 contiguous text(연속된 문장)을 가지고 있어서 model이 long-range information을 잘 학습하게끔 하는 이점이 있다.

논문에서 제시한 model은 transformer decoder로 이루어진 12개의 layer를 사용하였으며, 각 transformer는 768의 dimensional states와 12개의 attention heads를 가지며, 3072 dimension의 position-wise feed-forward-networkf를 가진다.

Pre-training 할 때 Adam optimizer를 사용하였으며, learning rate의 경우 0부터 2.5e-4까지 2000 step 동안 선형적으로 증가한 뒤 cosine 함수를 따라 0까지 감소하게끔 설정하였으며, sample당 512개의 token을 가졌으며, batch size는 64로 학습을 진행하였다.

또한, 40000번 merge 한 BPE(Byte-Pair Encoding)을 사용하였으며, activation function으로는 GELU(Gaussian Error Linear Unit)을 사용하였다.

눈에 띄는 점은 transformer 논문(Attention is all you need)에서 제안되었던 sinusoidal position embedding을 사용하지 않고 learned position embedding을 사용하였다고 한다.

Fine-tuningd을 진행할 때에도 unsupervised pre-training의 hyperparameter와 대부분 동일하지만, $p = 0.1$의 dropout을 추가하였고, learning rate : 6.25e-5, batch size : 32, 0.2%의 learning rate decay와 앞서 살펴본 auxiliary objective의 weight λ로 0.5를 주었다는 차이가 있다.

Supervised fine-tuning

NLP 분야의 다양한 task(natural language inference, question answering, semantic similarity, and text classification)에 대해 실험을 진행하였고, 해당 task에 적용한 dataset은 다음과 같다.

Table 1: A list of the different tasks and datasets used in our experiments.

Task	Datasets
Natural language inference	SNLI [5], MultiNLI [66], Question NLI [64], RTE [4], SciTail [25]
Question Answering	RACE [30], Story Cloze [40]
Sentence similarity	MSR Paraphrase Corpus [14], Quora Question Pairs [9], STS Benchmark [6]
Classification	Stanford Sentiment Treebank-2 [54], CoLA [65]

먼저, natural language inference를 살펴보자

Table 2: Experimental results on natural language inference tasks, comparing our model with current state-of-the-art methods. 5x indicates an ensemble of 5 models. All datasets use accuracy as the evaluation metric.

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

textual entailment라고 알려진 natural language inference(NLI) task는 pair of sentence를 읽고 두 sentence의 관계를 entailment(귀결), contradiction(모순), neutral(중립) 중 하나의 관계로 판단한다.

이 task를 위해 각각 Image caption(SNLI), 문서화된 음성, 소설과 정부 보고서(MNLI), 위키피디아 기사(QNLI), 과학시험(SciTail), 뉴스 기사(RTE) 도메인에서 추출한 데이터셋을 이용하였다.

그 결과, 데이터 크기가 가장 적은 RTE 데이터셋을 제외하고, 논문에서 제시하는 모델이 가장 좋은 성능을 보이는 것을 확인할 수 있다.

이번에는 Question answering and commonsense reasoning task에 대한 결과이다.

Table 3: Results on question answering and commonsense reasoning, comparing our model with current state-of-the-art methods.. 9x means an ensemble of 9 models.

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**

중고등학교 시험지의 영어 지문 및 질문으로 구성된 RACE, 주어진 문장들에 이어질 적합한 엔딩을 고르는 Story Cloze를 사용하며, 모든 지문에서 SOTA를 달성하였다. 또한 여러 문장이 주어진 Story Cloze에서 비약적인 성능 향상을 보이는 점에서 미루어 논문에서 제시하는 model이 long-range context를 잘 포착하고 있음을 확인할 수 있다.

마지막으로, Semantic Similarity과 Classification task에 관해 살펴보겠다.

Table 4: Semantic similarity and classification results, comparing our model with current state-of-the-art methods. All task evaluations in this table were done using the GLUE benchmark. (*mc*= Mathews correlation, *acc*=Accuracy, *pc*=Pearson correlation)

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**

먼저, 두 sentence 간 유사도를 구하는 Semantic Similarity task이다.

뉴스로부터 수집한 dataset인 MRPC와 QQP, STS-B dataset을 사용하였는데, QQP와 STS-B dataset에서 SOTA를 달성하였다.

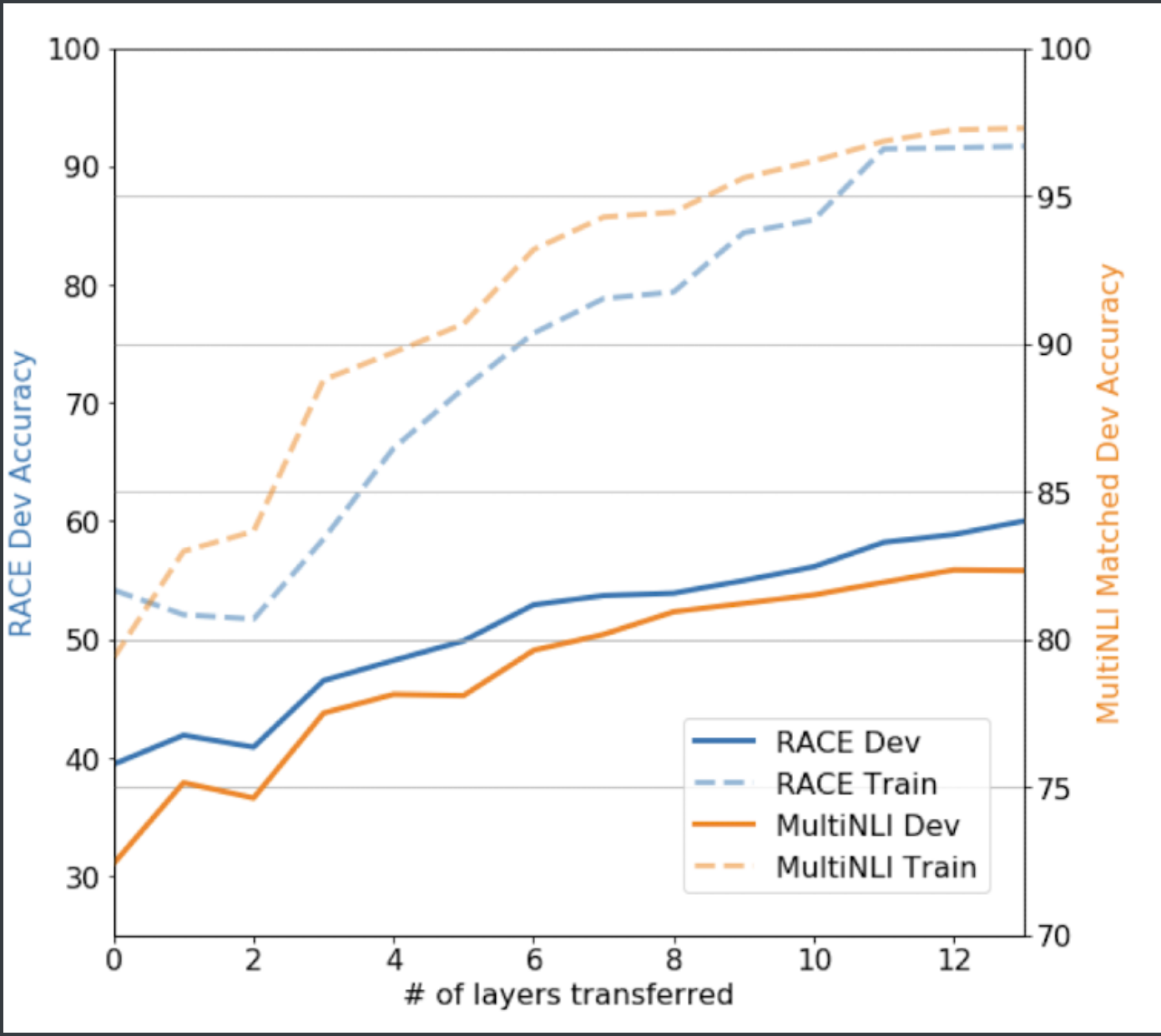
Classification task의 경우, 문장이 문법적으로 옳은지에 대한 전문가의 판단과 학습된 model의 linguistic bias에 대한 테스트를 포함하는 CoLA(The Corpus of Linguistic Acceptability) dataset과 표준 이진 분류 문제인 SST-2(The Stanford Sentiment Treebank) dataset을 사용하였으며, 두 dataset을 사용하였을 때 모두 SOTA를 달성하였다.

전반적으로, 논문에서 제시하는 모델은 12개의 dataset 중에서 9개에서 SOTA를 달성하였다. 또한 dataset의 size와 관계없이 좋은 성능을 내는 것을 확인할 수 있었다.

Analysis

Impact of number of layers transferred

저자들은 연구를 진행하면서 unsupervised pre-training에서 supervised target task로 transfer 하는 layer의 개수의 영향을 관찰하였다. 아래의 사진을 살펴보자.



위 그래프는 MNLI dataset과 RACE dataset에서, unsupervised pre-training에서 supervised target task로 transfer하는 layerd의 개수 당 model 성능을 나타낸 그래프이다.

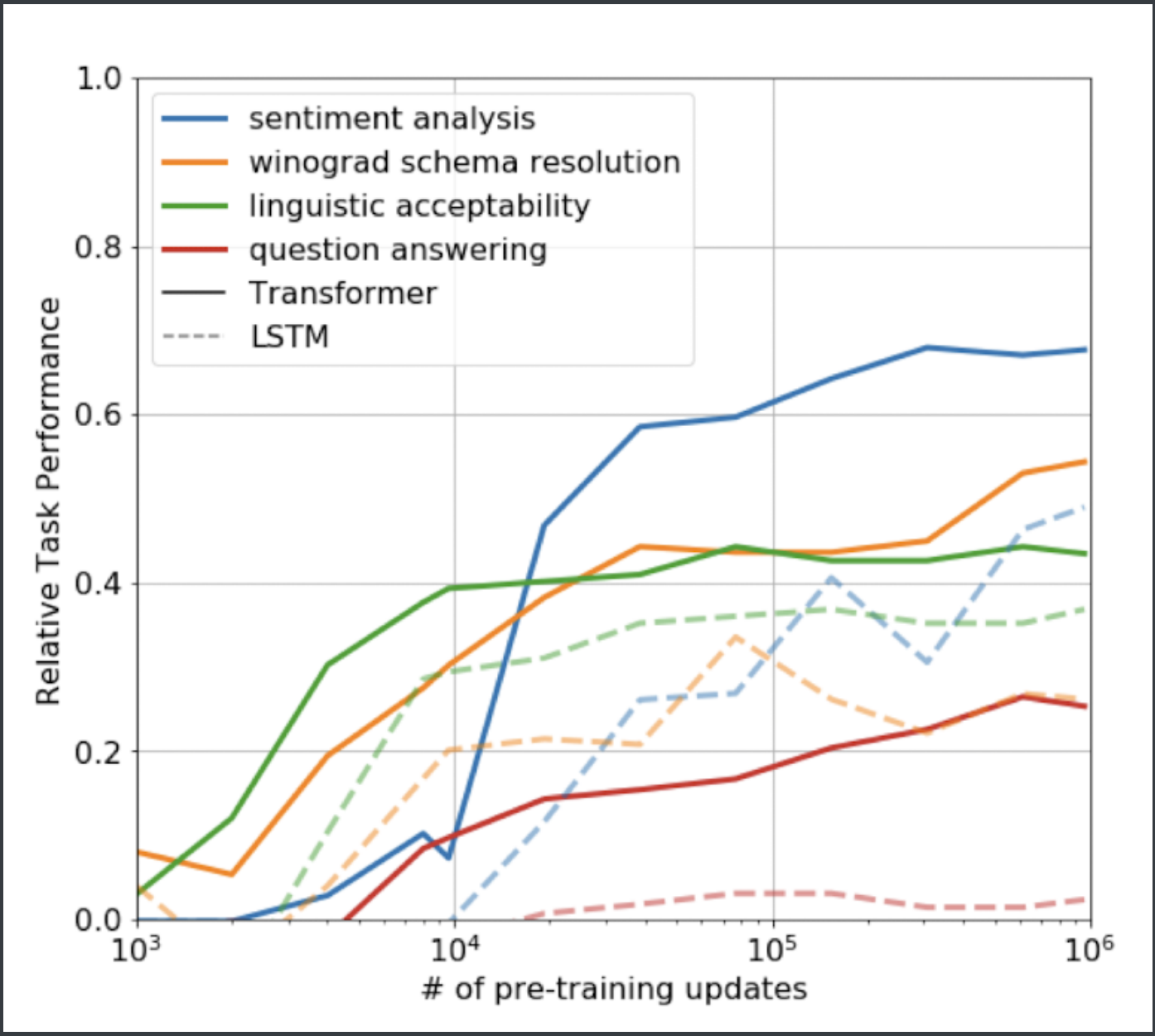
transfer하는 layer의 개수가 증가할수록 정확도가 향상하는 것을 볼 수 있는데, 이는 사전학습 모델의 각 layer가 target task에 적합한 기능을 학습했음을 의미한다.

Zero-shot Behaviors

논문에서는 transformer를 이용한 pre-training이 효과적인 이유를 찾고자 하였다. 그래서 다음과 같은 가설을 세운다.

- Underlying generative model(unsupervised pre-trained model)이 language model의 성능을 향상하기 위해서 다양한 task를 수행하는 방법을 학습한다.
- LSTM에 비해 Transformer의 attentional memory 가 전이하기에 적합하다.

저자들은 이러한 가설들을 검증하기 위해 supervised fine-tuning 없이 underlying generative model(unsupervised pre-trained model)만으로 각 task를 수행하는 실험을 하여 zero-shot performance를 측정하였다. 결과는 아래와 같다.



우선, 실선의 경우 transformer로 수행된 결과이고, 점선은 LSTM으로 수행된 결과이다.

transformer로 수행된 결과인 실선부터 살펴보겠다. pre-training 횟수가 늘어날수록 성능이 향상하는 것을 볼 수 있으며, **generative pre-training(생성적 사전 학습)이 다양한 task와 관련된 다양한 기능을 학습하였음을 알 수 있다.**

또한, LSTM으로 수행된 결과인 점선을 살펴보면, transformer로 동일한 작업을 수행한 결과에 비해 낮은 zero-shot performance를 보임을 확인할 수 있다. 논문에서는 이러한 이유로 **transformer의 inductive bias가 transfer를 돕는 것이**라고 추측한다.

(**Transformer의 경우 RNN, CNN에 비해 inductive bias가 부족하다. 이러한 inductive bias가 높을수록 작은 dataset에서도 학습 성능이 높아지긴 하지만, generalization(variance)은 낮아진다.** transformer의 상대적으로 낮은 inductive bias로 인해 RNN 계열의 LSTM보다 **robust 하게 동작**하기 때문에 보다 다양한 task에 적용해야 하는 transfer에서 강세를 보인다고 생각한다.)

Ablation studies

논문에서는 제시한 방법론들의 효과를 알아보기 위해 abalation study를 진행하였다. 결과는 아래와 같다.

Table 5: Analysis of various model ablations on different tasks. Avg. score is a unweighted average of all the results. (*mc*= Mathews correlation, *acc*=Accuracy, *pc*=Pearson correlation)

Method	Avg. Score	CoLA (mc)	SST2 (acc)	MRPC (F1)	STSB (pc)	QQP (F1)	MNLI (acc)	QNLI (acc)	RTE (acc)
Transformer w/ aux LM (full)	74.7	45.4	91.3	82.3	82.0	**70.3**	**81.8**	**88.1**	**56.0**
Transformer w/o pre-training	59.9	18.9	84.0	79.4	30.9	65.5	75.7	71.2	53.8
Transformer w/o aux LM	**75.0**	**47.9**	**92.0**	**84.9**	**83.2**	69.8	81.1	86.9	54.4
LSTM w/ aux LM	69.1	30.3	90.5	83.2	71.8	68.1	73.7	81.1	54.6

먼저, 저자들은 fine-tuning의 objective function $L_2(C)$에 auxiliary objective(보조 목적함수)로 language modeling을 추가하는 것에 대한 측정을 시도하였다. 그 결과, **fine-tuning시 auxiliary objective(보조 목적함수)를 추가하는 것은 큰 데이터셋에는 효과적이지만, 작은 데이터셋에는 그렇지 않음을 확인할 수 있다.**

이어서, LSTM과 비교하여 Transformer의 효과를 분석하였다. 단일 layer의 2048 unit **LSTM과 비교하였을 때 transformer가 더 좋은 성능을 보이는 것을 확인할 수 있다.**

마지막으로, pre-training 없이 target task에 바로 학습한 결과도 도출하였다. 그 결과, **pre-training 없이는 모든 task에서 성능이 저하하였음을 확인할 수 있다.**