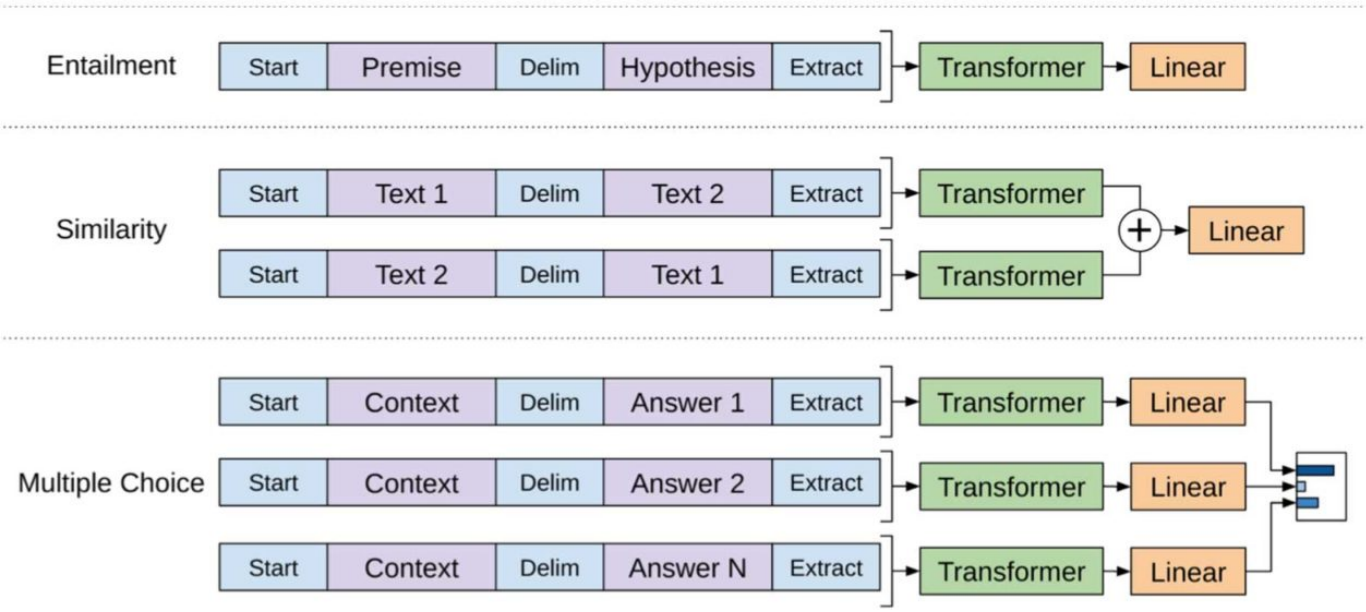
Generalized Language Models: ULMFiT & OpenAI GPT

April 24, 2019 by Lilian Weng



EDITOR'S NOTE: Generalized Language Models is an extensive four-part series by Lillian Weng of OpenAI.

- Part 1: CoVe, ELMo & Cross-View Training
- Part 2: ULMFiT & OpenAI GPT
- Part 3: BERT & OpenAI GPT-2
- Part 4: Common Tasks & Datasets

Do you find this in-depth technical education about language models and NLP applications to be useful? *Subscribe below to be updated when we release new relevant content.*

In this post, we will continue our deep-dive on the pre-trained language models:

- ULMFiT
- OpenAI GPT
  - Transformer Decoder as Language Model
  - BPE
  - Supervised Fine-Tuning

ULMFiT

The idea of using generative pretrained LM + task-specific fine-tuning was first explored in ULMFiT (Howard & Ruder, 2018), directly motivated by the success of using ImageNet pre-training for computer vision tasks. The base model is AWD-LSTM.

2) *Target task LM fine-tuning*: ULMFiT proposed two training techniques for stabilizing the fine-tuning process. See below.

- **Discriminative fine-tuning** is motivated by the fact that different layers of LM capture different types of information (see discussion here). ULMFiT proposed to tune each layer with different learning rates, $\eta^1, \ldots, \eta^\ell, \ldots, \eta^L$, where η is the base learning rate for the first layer, η^ℓ is for the ℓ-th layer and there are L layers in total.
- **Slanted triangular learning rates (STLR)** refer to a special learning rate scheduling that first linearly increases the learning rate and then linearly decays it. The increase stage is short so that the model can converge to a parameter space suitable for the task fast, while the decay period is long allowing for better fine-tuning.

3) *Target task classifier fine-tuning*: The pretrained LM is augmented with two standard feed-forward layers and a softmax normalization at the end to predict a target label distribution.

- **Concat pooling** extracts max-polling and mean-pooling over the history of hidden states and concatenates them with the final hidden state.
- **Gradual unfreezing** helps to avoid catastrophic forgetting by gradually unfreezing the model layers starting from the last one. First the last layer is unfrozen and fine-tuned for one epoch. Then the next lower layer is unfrozen. This process is repeated until all the layers are tuned.

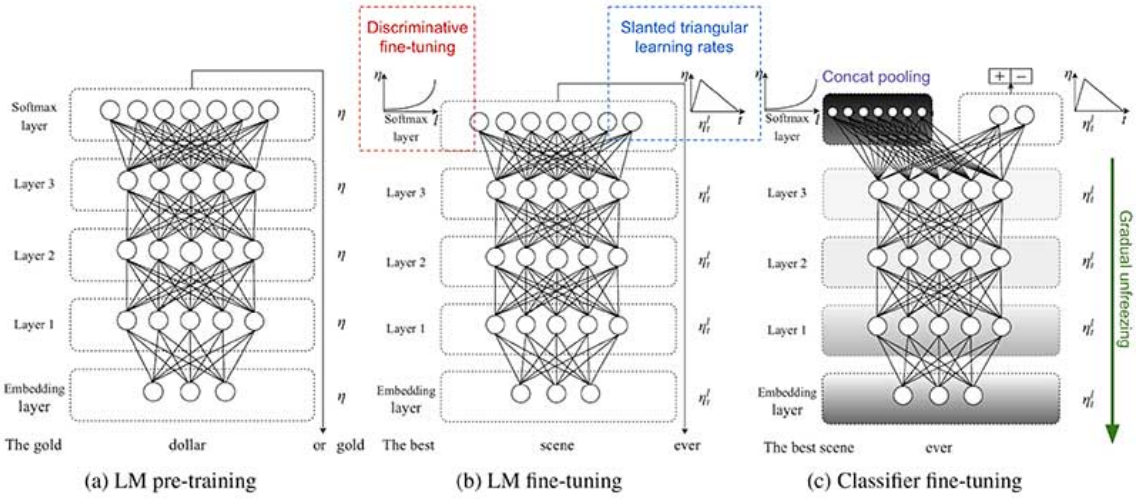


Fig. 1. Three training stages of ULMFiT. (Image source: original paper)

OpenAI GPT

Following the similar idea of ELMo, OpenAI **GPT**, short for **Generative Pre-training Transformer** (Radford et al., 2018), expands the unsupervised language model to a much larger scale by training on a giant collection of free text corpora. Despite of the similarity, GPT has two major differences from ELMo.

1. The model architectures are different: ELMo uses a shallow concatenation of independently trained left-to-right and right-to-left multi-layer LSTMs, while GPT is a multi-layer transformer decoder.
2. The use of contextualized embeddings in downstream tasks are different: ELMo feeds embeddings into models customized for specific tasks as additional features, while GPT fine-tunes the same base model for all end tasks.

input sentence rather than two separate source and target sequences.

This model applies multiple transformer blocks over the embeddings of input sequences. Each block contains a masked *multi-headed self-attention* layer and a *pointwise feed-forward* layer. The final output produces a distribution over target tokens after softmax normalization.

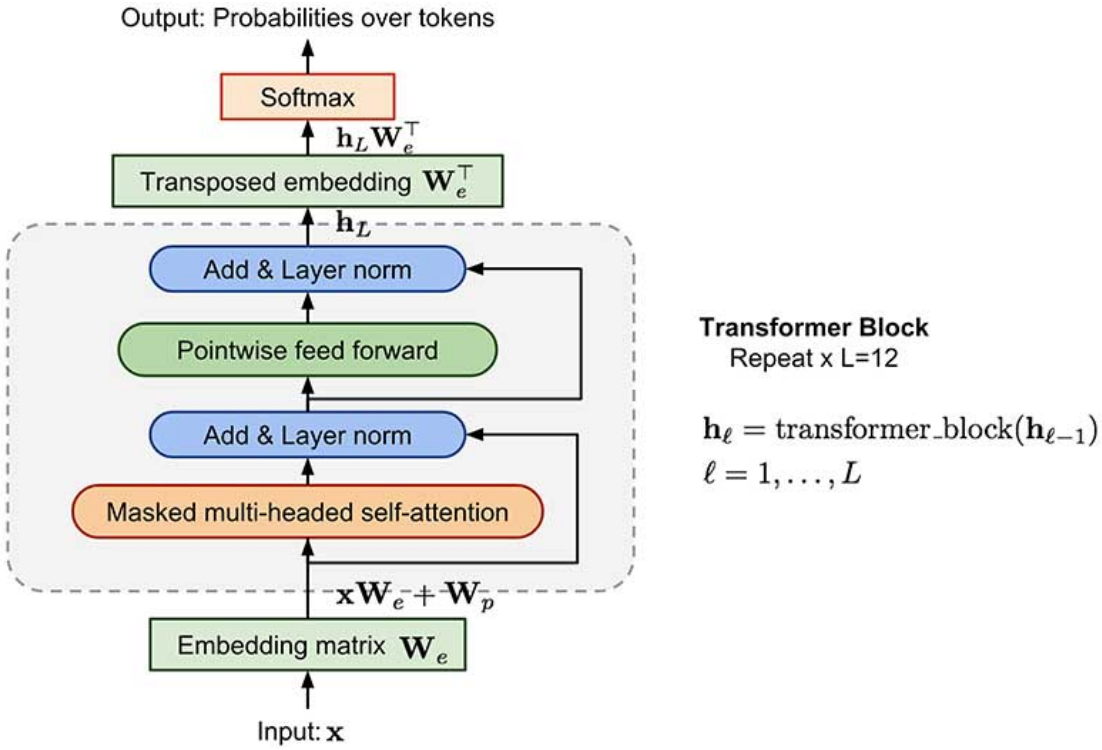


Fig. 2. The transformer decoder model architecture in OpenAI GPT.

The loss is the negative log-likelihood, same as ELMo, but without backward computation. Let's say, the context window of the size k is located before the target word and the loss would look like:

$$\mathcal{L}_{\text{LM}} = -\sum_i \log p(x_i \mid x_{i-k}, \ldots, x_{i-1})$$

BPE

Byte Pair Encoding (BPE) is used to encode the input sequences. BPE was originally proposed as a data compression algorithm in 1990s and then was adopted to solve the open-vocabulary issue in machine translation, as we can easily run into rare and unknown words when translating into a new language. Motivated by the intuition that rare and unknown words can often be decomposed into multiple subwords, BPE finds the best word segmentation by iteratively and greedily merging frequent pairs of characters.

Supervised Fine-Tuning

The most substantial upgrade that OpenAI GPT proposed is to get rid of the task-specific model and use the pre-trained language model directly!

Let's take classification as an example. Say, in the labeled dataset, each input has n tokens, $\mathbf{x} = (x_1, \ldots, x_n)$, and one label y. GPT first processes the input sequence $\mathbf{x}$ through the pre-trained transformer decoder and the last layer output for the last token x_n is $\mathbf{h}_L^{(n)}$. Then with only one new trainable weight matrix $\mathbf{W}_y$, it can predict a distribution over class labels.

$$P(y \mid x_1, \ldots, x_n) = \text{softmax}(\mathbf{h}_L^{(n)}\mathbf{W}_y)$$

The loss is to minimize the negative log-likelihood for true labels. In addition, adding the LM loss as an auxiliary loss is found to be beneficial, because:

- (1) it helps accelerate convergence during training and
- (2) it is expected to improve the generalization of the supervised model.

$$\mathcal{L}_{\text{cls}} = \sum_{(\mathbf{x},y)\in\mathcal{D}} \log P(y \mid x_1, \ldots, x_n) = \sum_{(\mathbf{x},y)\in\mathcal{D}} \log \text{softmax}(\mathbf{h}_L^{(n)}(\mathbf{x})\mathbf{W}_y)$$

$$\mathcal{L}_{\text{LM}} = -\sum_i \log p(x_i \mid x_{i-k}, \ldots, x_{i-1})$$

$$\mathcal{L} = \mathcal{L}_{\text{cls}} + \lambda\mathcal{L}_{\text{LM}}$$

With similar designs, no customized model structure is needed for other end tasks (see Fig. 2). If the task input contains multiple sentences, a special delimiter token ($) is added between each pair of sentences. The embedding for this delimiter token is a new parameter we need to learn, but it should be pretty minimal.

For the sentence similarity task, because the ordering does not matter, both orderings are included. For the multiple choice task, the context is paired with every answer candidate.

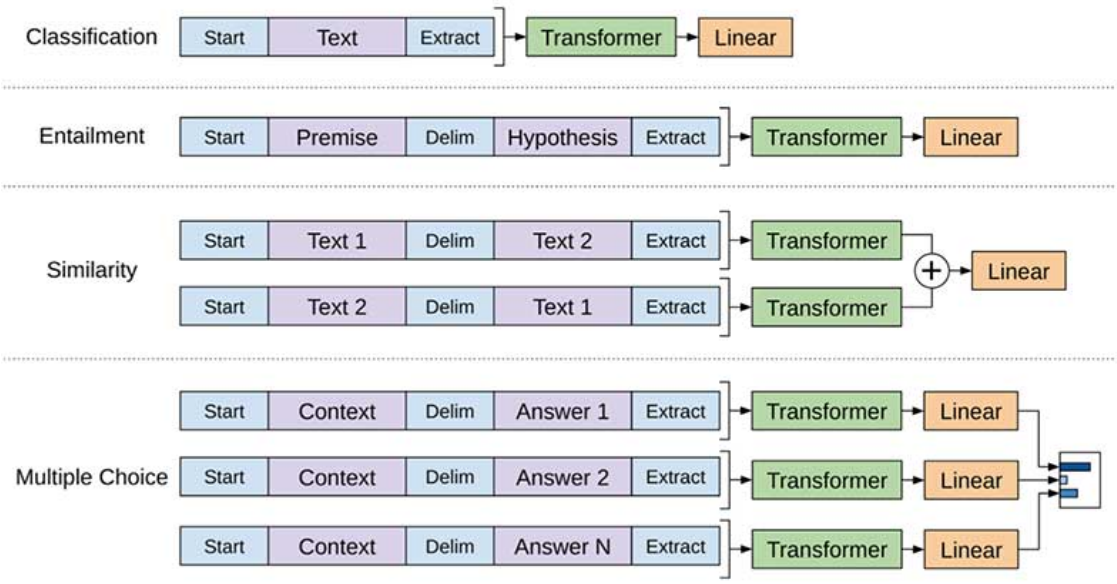


Fig. 3. Training objects in slightly modified GPT transformer models for downstream tasks. (Image source: original paper)

Summary: It is super neat and encouraging to see that such a general framework is capable to beat SOTA on most language tasks at that time (June 2018). At the first stage, generative pre-training of a language model can absorb as much free text as possible. Then at the second stage, the model is fine-tuned on specific tasks with a small labeled dataset and a minimal set of new parameters to learn.

One limitation of GPT is its uni-directional nature — the model is only trained to predict the future left-to-right context.

Enjoy this article? Sign up for more AI and NLP updates.

We'll let you know when we release more in-depth technical education.

Email Address *

Name *

First

Last

Company *

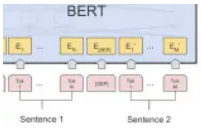
What areas of AI research are you interested in? Select all that apply *

- ☐ Natural Language Processing (NLP)
- ☐ Chatbots & Conversational AI
- ☐ Computer Vision
- ☐ Ethics & Safety
- ☐ Robotics
- ☐ Machine Learning
- ☐ Deep Learning
- ☐ Reinforcement Learning
- ☐ Generative Models
- ☐ Other (Please Describe Below)

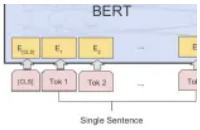
What is your biggest challenge with AI research? *

SUBMIT

Related

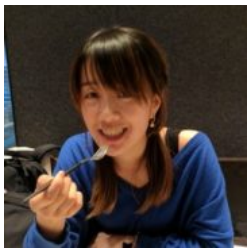


Generalized Language Models: BERT & OpenAI GPT-2



Generalized Language Models: Common Tasks & Datasets

What Every NLP Engineer Needs to Know About Pre-Trained Language Models



About Lilian Weng

Lilian Weng is on the Robotics team at OpenAI. She writes code, reads papers, does research on deep learning models, and works on physical machines.

Leave a Reply

Your email address will not be published. Required fields are marked *

Comment *

Name

Email

Website

POST COMMENT

Search the site ...

Learn Applied AI

POPULAR ARTICLES

NeurIPS 2021 – 10 Papers You Shouldn't Miss

A Guide To Knowledge Graphs

Why Graph Theory Is Cooler Than You Thought

10 Leading Language Models For NLP In 2021

Pretrain Transformers Models in PyTorch Using Hugging Face Transformers

More Articles

TOPICS

Bots
Brands
Business
China
Commerce
Computer Vision
Conversational AI
Customer Service
Cybersecurity
Data Science & Engineering
Design
Education
Ethics & Safety
Finance
Gaming
Healthcare
HR & Recruiting
Infrastructure
Leadership & Management
Manufacturing
Marketing
Natural Language Processing
Reinforcement Learning
Research
Retail & CPG
Society
Technical Guide