NLP Pretraining:
Deeply Encoded Representations

DATA/COMPSCI 182 Deep Learning
Lecture 18 11/07/2024

(NLP) Pretraining: Recap

- The **upstream**
- Embeddings
- Work (token) embeddings
- Word2vec
- Today
  - Continue on the **upstream** (pretraining)
  - But **richer representations**, that are **task agnostic**
    - More *easily support* **downstream** tasks

In the small town of Willowbrook, Sam and Lucy visited the newly opened Oakridge Museum. Sam, an art lover, was thrilled, his eyes sparkling as he admired the historic paintings and artifacts. "This is amazing, Lucy!" he beamed. Lucy, however, wasn't impressed. "This place feels ancient and boring," she grumbled.

They passed a statue of the town's founder, William Oakridge, and saw artifacts from the Civil War. Sam eagerly pointed out details, but Lucy just sighed, glancing at her watch. Sam, engrossed, paid no mind, enjoying every bit of history the museum had to offer.

GPT (2018)

GPT

- The “duality”
  - **Generative pre-training**
  - **Discriminative fine-tuning**
- Distinguishing feature: **Task-aware** transformations
- Validates on 12 NLP tasks
  - Sentiment Analysis
  - Question Answering (QA)
  - Natural Language Inference (NLI)
  - Textual Entailment
  - Classification
  - Entity & Relationships Extraction
  - …

Framework

- Leveraging **more than** word level information
  - What's the optimization obejctive ?
    - To achieve good transfer
- A **semi-supervised** approach
  - Unsupervised pretraining
  - Supervised fine tuning
- **GPT goal:** learn a **universal representation** that **transfers with minimal effort** to specific tasks
- Two-stage training procedure
  - Use a **language modeling objective** on the unlabeled data to learn the **initial parameters** of a neural network model
  - Subsequently, **adapt these parameters** to a target task using the **corresponding supervised objective**
- Evaluate on four types of language understanding tasks – natural language inference, question answering, semantic similarity, and text classification
- **Transformer** based

Framework

- **Unsupervised pre-training**

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

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

- $U = \{u_{-k}, \ldots, u_{-1}\}$: context vector of tokens
- W_e : Token embedding matrix
- W_p: Position embedding matrix

- **Supervised fine-tuning**

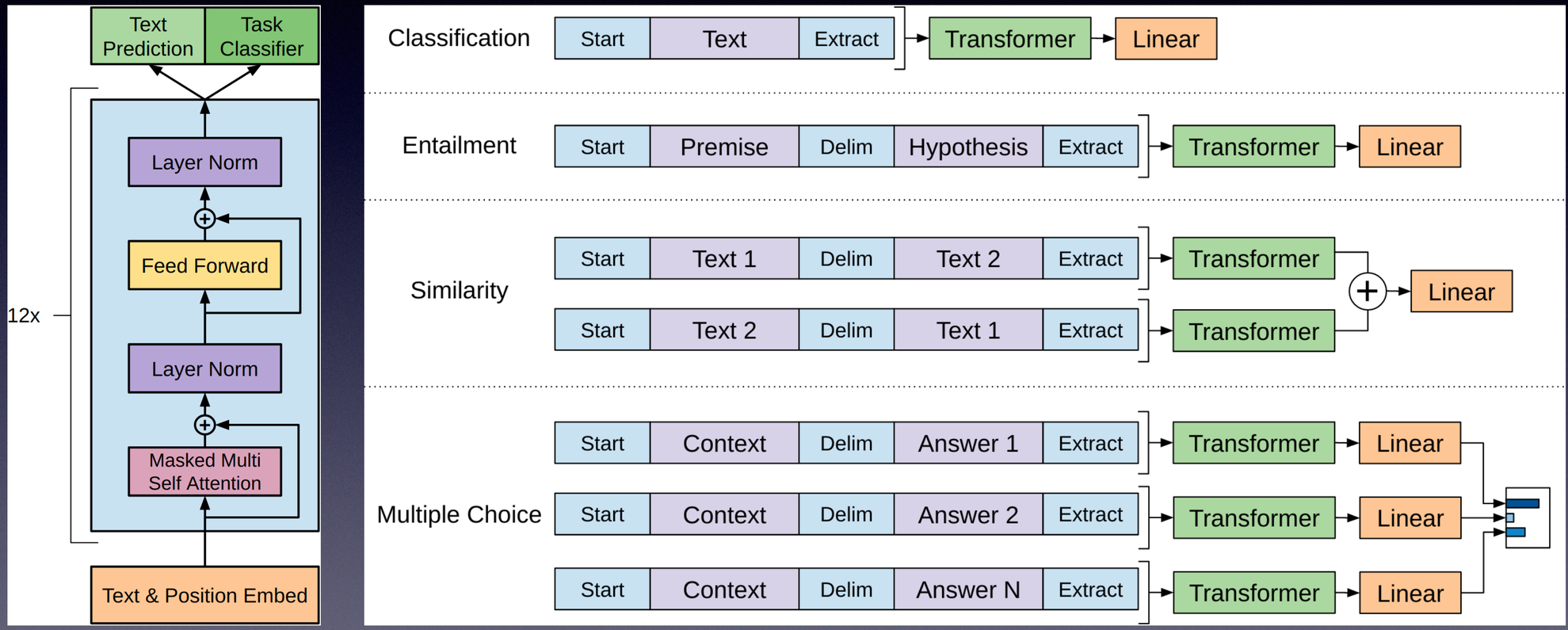
$$P(y|x^1, \ldots, x^m) = \texttt{softmax}(h_l^m W_y)$$

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

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

- The **only extra parameters** required for fine tuning are ***Wy***, and delimeter token embeddings

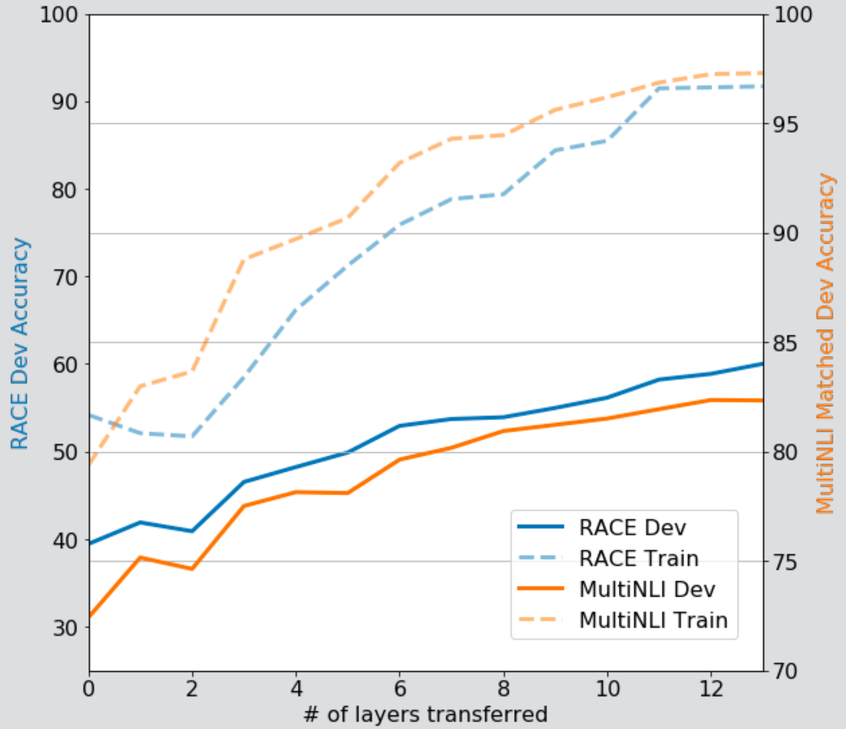
Architecture

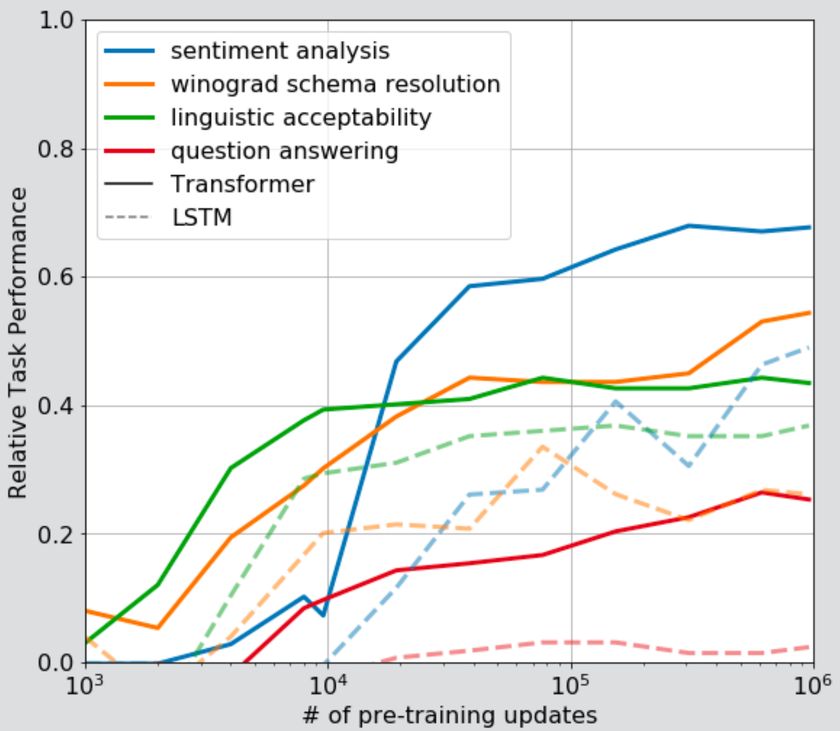


Details

- Unsupervised pre-training
  - BooksCorpus dataset for training the language model
    - Contains over 7,000 unique unpublished books from a variety of genres
    - Contains long stretches of contiguous text !
      - Which allows the generative model to learn to condition on long-range information.
- Model specs
  - 12-layer decoder-only transformer
  - Masked self-attention heads (768 dimensional states and 12 attention heads)
  - Position-wise feed-forward: 3072 dim insser states
  - Adam optimization, Max learning rate of 2.5e-4
  - 100 epochs, mini batches of 64, sequences of 512 contiguous tokens
  - Byte pair encoding BPE
  - L2 regularization

Impact: of Number of Layers Transferred





Impact: Number of Layers

Zero-shot: Number of pre-training updates

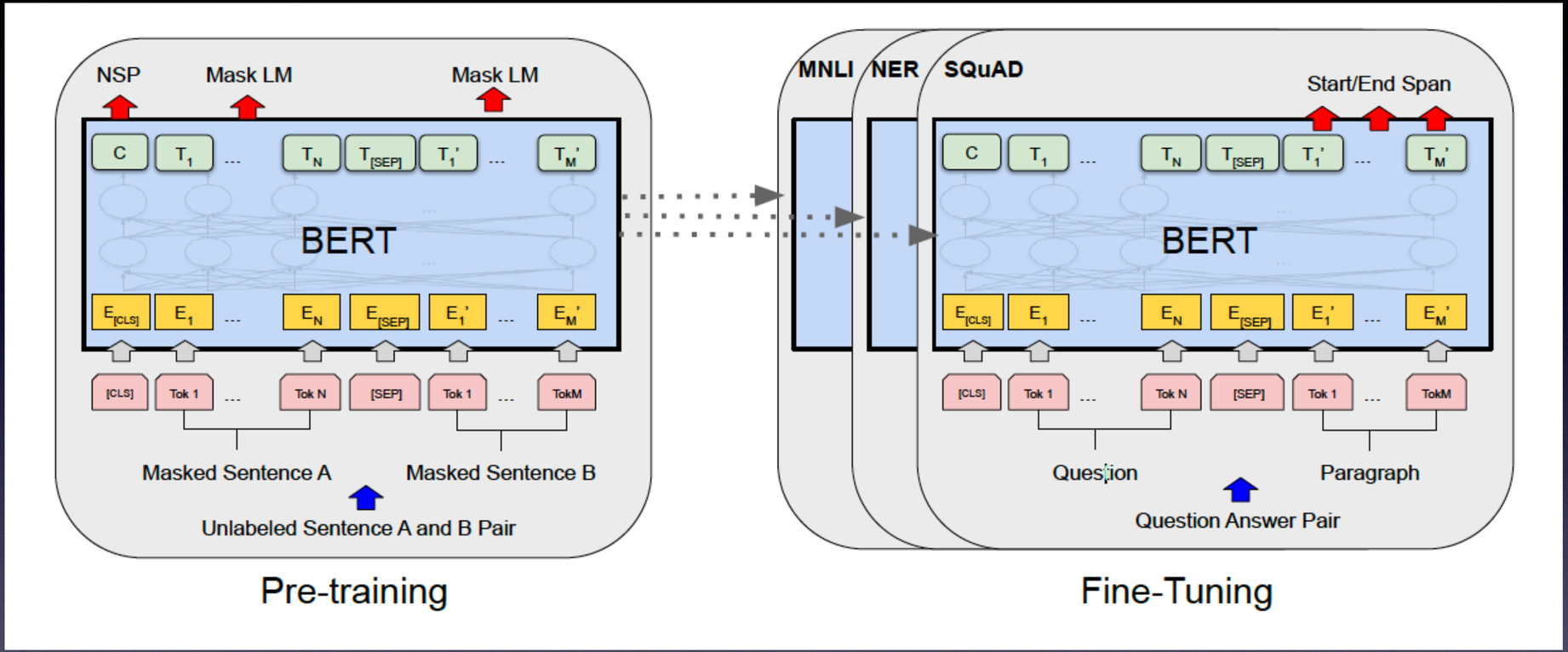
BERT (2019)

BERT

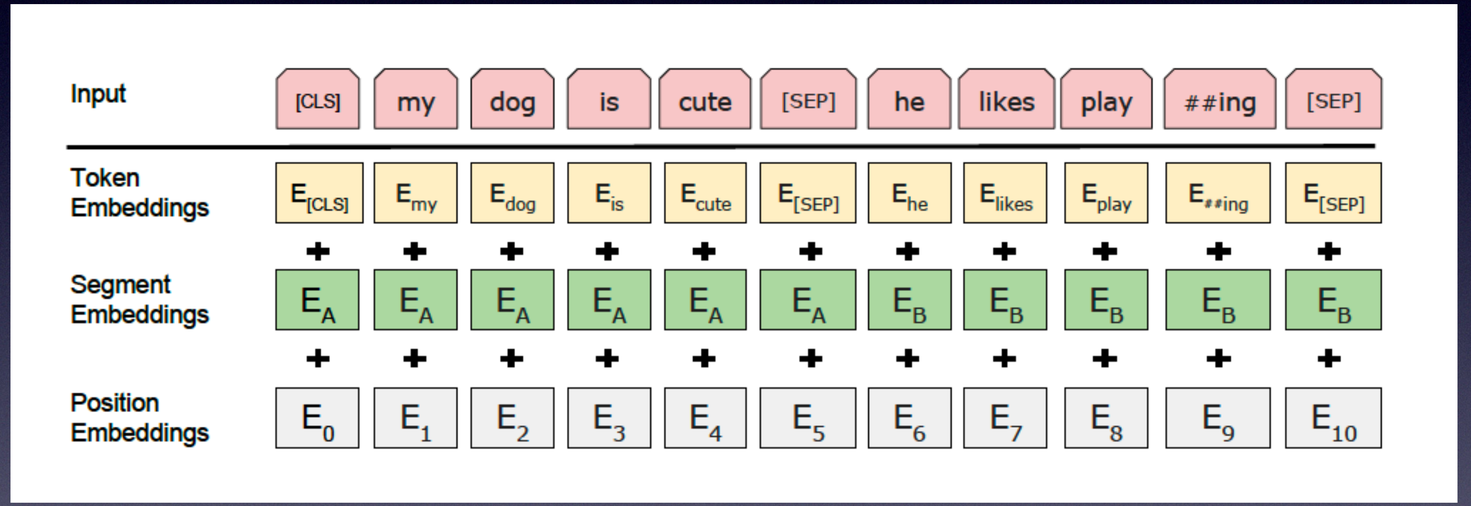
- Bidirectional !
- Designed to pretrain **deep bidirectional representations** from unlabeled text
  - by jointly conditioning on **both left and right context** in all layers.
- the pre-trained BERT model can be fine-tuned with *just one additional* output layer
  - to create state-of-the-art models for a wide range of tasks
    - such as question answering and language inference
    - **without** substantial taskspecific architecture modifications.
- Masked Language Model
- Deep bidirectionally encoded representation —> Minimal effort for downstream tasks
- Code: https://github.com/google-research/bert

Architecture

- Pre-training
  - Unsupervised i.e., trained on unlabeled data
- Fine-tuning
  - Supervised, on labeled data
  - Carry over the parameters from pretraining !
- [CLS] and [SEP]
- Unified architecture, across pretraining and fine-tuning
  - **Q: Did we see that in GPT ?**



BERT Input Representation



- Sum of three embeddings

BERT Architecture

- Multilayer Transformer ENCODER
  - Based on original implementation of Vaswani et al 2018 (Attentionis all you need)
    - Tensor2tensor library

- BERT Transformer
  - Layers L = 12
  - Hidden size H = 768
  - Self-Attention Heads A = 12
  - BERTBASE (L=12, H=768, A=12, Total Parameters=110M)
  - BERTLARGE (L=24, H=1024, A=16, Total Parameters=340M)
- BERTBASE was chosen to have the same model size as OpenAI GPT for comparison purposes
- Critically, however, the BERT Transformer uses bidirectional self-attention, while the GPT Transformer uses constrained self-attention where every token can only attend to context to its left.

Input/Output Representations

- To make BERT handle a variety of down-stream tasks, our input representation is able to unambiguously represent both a single sentence and a pair of sentences
  - (e.g., Question, Answer) in one token sequence.
- Throughout this work, a “sentence” can be an arbitrary span of contiguous text, rather than an actual linguistic sentence.
- Wordpiece
  - 30K
  - CLS and SEP

Pretraining BERT

- TWO unsupervised tasks
- Task #1: Masked LM
  - Standard conditional models : bidirectional —> “see itself”
  - Mask at random, 15%, [MASK]
- Task #2: Next Sentence prediction (NSP)
- Pretraining data
  - BookCorpus: 800M words
  - English Wikipedia: 2500M words

Fine-tuning BERT

- Relies on self-attention
- Simply fine-tune ALL parameters on specific task
- Relatively efficient: few hours on GPU (then)

Evaluation

- Accuracy: on GLUE, SQuAD v1.0 and 2.0, SWAG

- Effection of Pretraining: Ablation
  - No NSP: No next sentence prediction (but MLM employed)
  - LTR & No NSP: (left-to-right only, no MLM)
    - Like OpenAI GPT !
    - +BiLSTM: Randomly initialized bidirectional LSTM on top

Tasks	Dev Set MNLI-m (Acc)	QNLI (Acc)	MRPC (Acc)	SST-2 (Acc)	SQuAD (F1)
$\text{BERT}_{\text{BASE}}$	84.4	88.4	86.7	92.7	88.5
No NSP	83.9	84.9	86.5	92.6	87.9
LTR & No NSP	82.1	84.3	77.5	92.1	77.8
+ BiLSTM	82.1	84.1	75.7	91.6	84.9

- Effect of model size
  - L: Layers H: Hidden units A: Attention heads
  - LM (ppl): Language Model Perplexity

Hyperparams			Dev Set Accuracy			
#L	#H	#A	LM (ppl)	MNLI-m	MRPC	SST-2
3	768	12	5.84	77.9	79.8	88.4
6	768	3	5.24	80.6	82.2	90.7
6	768	12	4.68	81.9	84.8	91.3
12	768	12	3.99	84.4	86.7	92.9
12	1024	16	3.54	85.7	86.9	93.3
24	1024	16	3.23	86.6	87.8	93.7

Feature based approach

- All of the BERT results presented so far have used the fine-tuning approach
  - where a simple classification layer is added to the pre-trained model
  - and all parameters are jointly fine-tuned on a downstream task
- But .. not all tasks can be easily represented by a transformer encoder architecture
  - And there are major computational benefits in training an expensive representation once
- **Feature-based:** extract activations from one/more layers *without* fine-tuning (any parameters)
  - Contextual embeddings to 2 layer (randomly initialized) BiLSTM on top
- BERT is effective for both fine-tuning and feature-based approaches !

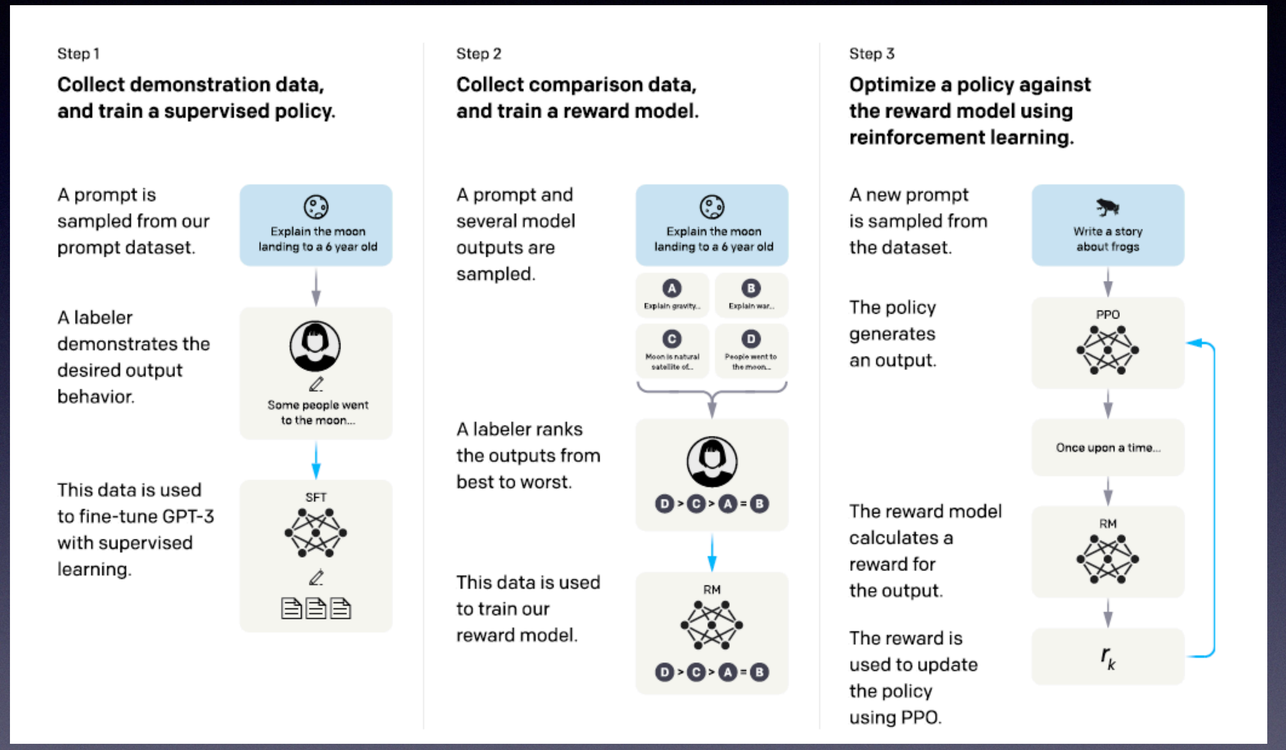
System	Dev F1	Test F1
ELMo (Peters et al., 2018a)	95.7	92.2
CVT (Clark et al., 2018)	-	92.6
CSE (Akbik et al., 2018)	-	**93.1**
Fine-tuning approach		
$\text{BERT}_{\text{LARGE}}$	96.6	92.8
$\text{BERT}_{\text{BASE}}$	96.4	92.4
Feature-based approach ($\text{BERT}_{\text{BASE}}$)		
Embeddings	91.0	-
Second-to-Last Hidden	95.6	-
Last Hidden	94.9	-
Weighted Sum Last Four Hidden	95.9	-
Concat Last Four Hidden	96.1	-
Weighted Sum All 12 Layers	95.5	-

InstructGPT (2022)

Core model driving ChatGPT

Fine-tuning GPT-3 —> InstructGPT

Reinforcement Learning from Human Feedback (RLHF) Pipeline



- **Supervised Fine-Tuning**: Initial training with human-labeled examples to shape model behavior.
- **Reward Model Training**: Human-labeled rankings train a reward model to evaluate response quality.
- **Policy Optimization**: Proximal Policy Optimization (PPO) uses reward feedback to refine the model's responses iteratively

Model Development

- **RLHF Fine-Tuning**: Used Reinforcement Learning from Human Feedback to guide model training.
- **Human-Labeled Data**: Included human demonstrations and ranked comparisons to train a reward model.
- **Reward Model for Feedback**: Developed a reward model that scores outputs based on human preference.
- **PPO Optimization**: Applied Proximal Policy Optimization to maximize alignment with human feedback.
- **Parameter Efficiency**: Achieved high performance with smaller models by optimizing training on task-specific feedback rather than solely scaling up parameters.
- **Safety-Focused Development**: Added steps in the training pipeline to reduce harmful and inaccurate outputs