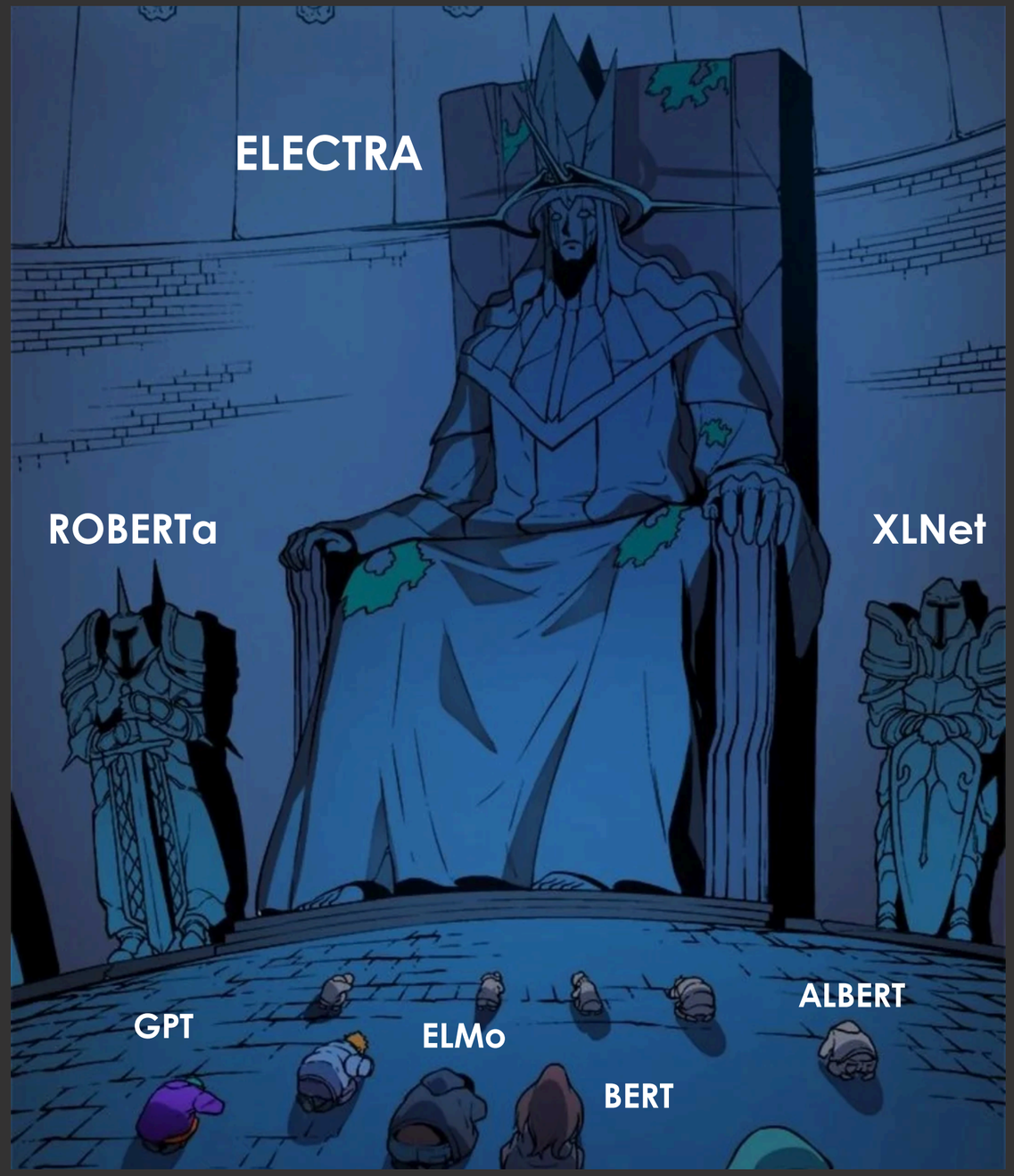
ELECTRA

PRE-TRAINING TEXT ENCODERS AS DISCRIMINATORS RATHER THAN GENERATORS

Authors: Kevin Clark, Minh-Thang Luong, Quoc V. Le, Christopher D. Manning

Google Brain
&
Standford University

Dipen Prajapati


ELECTRA
ROBERTa
XLNet
GPT
ELMo
BERT
ALBERT

COMPARISON

Feature	ELECTRA	BERT
Architecture	Transformer (Replaced Token Detection - RTD)	Transformer (Masked Language Model - MLM)
Efficiency	Most efficient (uses all tokens for learning)	Moderate (less efficient with MLM)
Speed	Fastest (efficient training)	Slower (due to large model sizes)
Size Comparison	ELECTRA-Small (~14M params) outperforms BERT-Base (~110M params)	BERT-Base (~110M params), BERT-Large (~340M params)
Use Cases	Best for fine-tuning on NLP tasks	General NLP tasks, widely used
Key Differences from BERT	Learns by identifying replaced tokens (RTD) rather than predicting original tokens from masked tokens (MLM). The discriminator learns from all tokens, making training more efficient.	Learns by predicting original tokens from masked tokens (MLM). Only masked tokens are used for learning.

OUTLINE

- INTRODUCATION
- MODEL ARCHITECTURE AND METHOD
- EXPERIEMENTS
- RESULTS
- CONCLUSION

INTRODUCTION

- **ELECTRA** stands for Efficiently Learning an Encoder that Classifies Token Replacements Accurately.
- pre-trained transformer-based model using self-supervised learning method for NLP.
- more efficient than BERT, XLNet, and RoBERTa.
- Uses a GAN-like approach to detect real vs. fake tokens.
- Requires less computation while achieving better accuracy.
- Faster inference and smaller model size make it great for deployment.

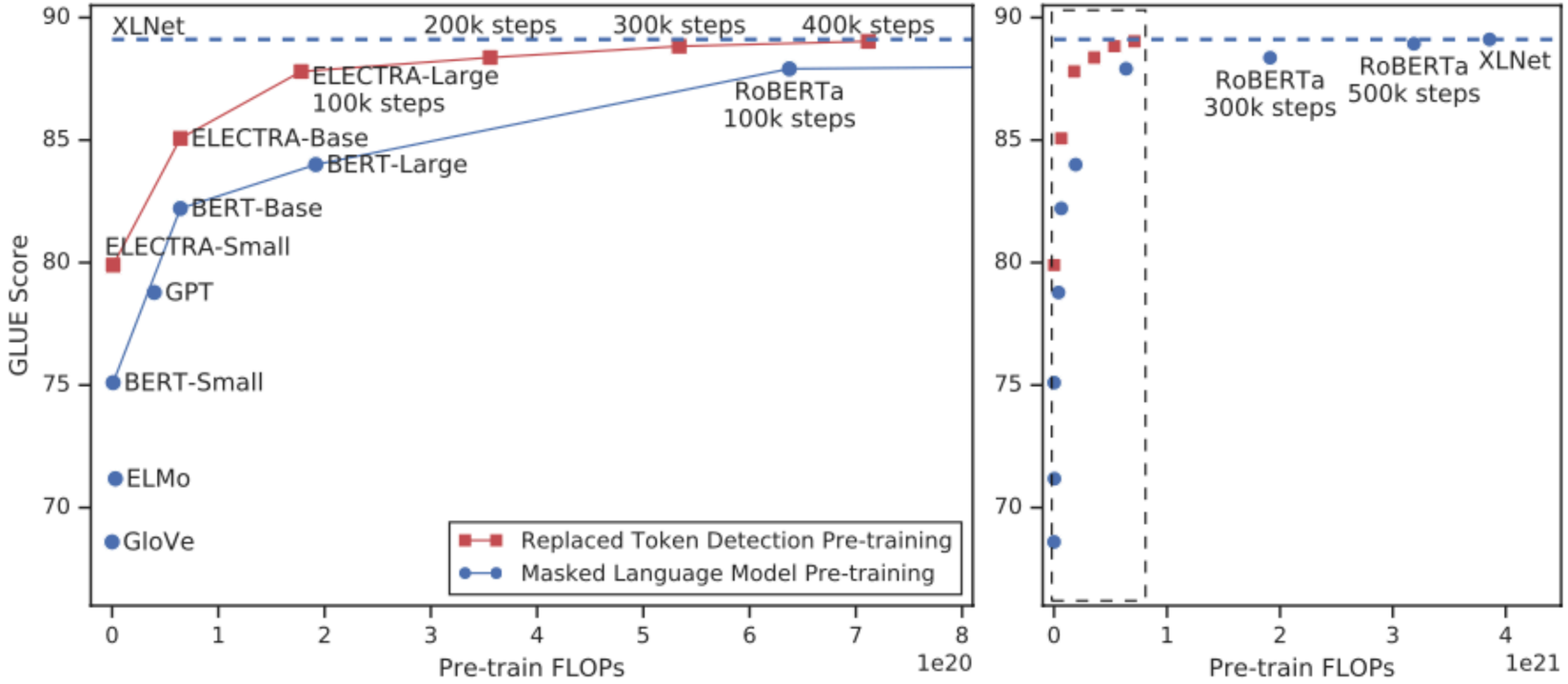
Approach

- Instead of masking, ELECTRA replaces tokens with fake ones.
- Uses Replaced Token Detection (RTD) to spot real vs. fake tokens.
- Learns from all tokens, unlike MLM models.
- Performs well on a single GPU at small scale.
- Achieves state-of-the-art results on SQuAD 2.0 at large scale.

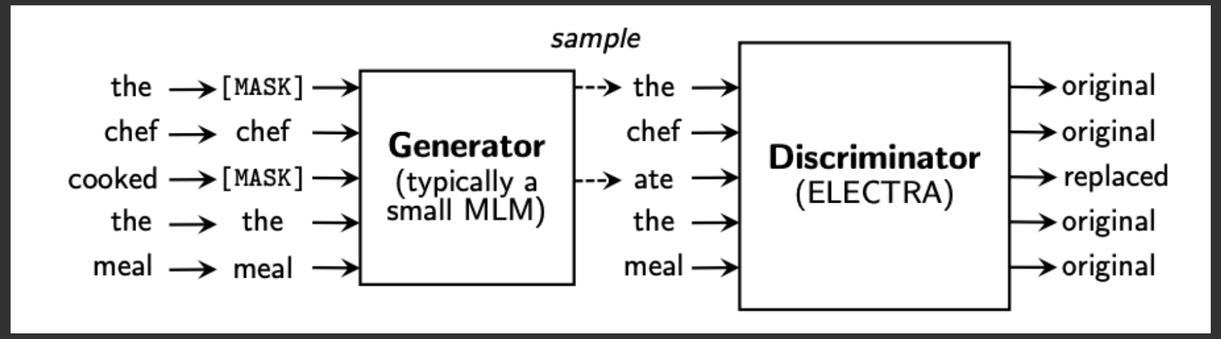
Motivation

- BERT and XLNet use Masked Language Modeling (MLM).
- Only 15% of tokens are masked, making training inefficient.
- BERT's [MASK] tokens cause a mismatch during fine-tuning.
- A better training method was needed for higher efficiency.

COMPARISON OF MODELS



MODEL ARCHITECTURE AND METHOD



Process Overview

- **Generator & Discriminator Architecture:**
  - **Generator:** A small masked language model (MLM) trained jointly with the discriminator.
  - **Discriminator:** Identifies whether tokens are “real” (from data) or “fake” (generated).
  - Unlike GANs(Generative Adversarial Networks), the generator is trained using maximum likelihood estimation (MLE) instead of adversarial training.
  - After pre-training, only the discriminator is fine-tuned on downstream tasks and generator is discarded after training process.

MODEL ARCHITECTURE AND METHOD

Generator- Masked Language Modeling(small BERT)

- **Input:** Sequence of tokens X = [X1, ..., Xn].
- **Masked Tokens:** A random set of positions are masked (m = [m1, ..., mk]), creating xmasked.
- **Prediction:** Generator predicts original tokens at masked positions.

$$\mathcal{L}_{\text{MLM}}(\boldsymbol{x}, \theta_G) = \mathbb{E}\left(\sum_{i \in \boldsymbol{m}} -\log p_G(x_i | \boldsymbol{x}^{\text{masked}})\right)$$

Discriminator- Token Classification

- **Input:** Corrupted sequence Xcorrupt, where masked tokens are replaced with generator samples.
- **Objective:** Predict whether each token in Xcorrupt is real or generated(fake).

$$\min_{\theta_G, \theta_D} \sum_{\boldsymbol{x} \in \mathcal{X}} \mathcal{L}_{\text{MLM}}(\boldsymbol{x}, \theta_G) + \lambda \mathcal{L}_{\text{Disc}}(\boldsymbol{x}, \theta_D)$$

Combined Loss Function (Optimization)

- The discriminator loss is not back-propagated through the generator.
- After pre-training, the generator is discarded, and the discriminator is fine-tuned for downstream tasks.

$$\mathcal{L}_{\text{Disc}}(\boldsymbol{x}, \theta_D) = \mathbb{E}\left(\sum_{t=1}^{n} -\mathbb{1}(x_t^{\text{corrupt}} = x_t) \log D(\boldsymbol{x}^{\text{corrupt}}, t) - \mathbb{1}(x_t^{\text{corrupt}} \neq x_t) \log(1 - D(\boldsymbol{x}^{\text{corrupt}}, t))\right)$$

MODEL ARCHITECTURE AND METHOD

How ELECTRA Differs from GANs

Aspect	GAN	ELECTRA
Training Method	Adversarial Training	Maximum Likelihood Estimation (MLE)
Objective	Generate Realistic Data	Detect Replaced Tokens
Gradient Flow	Discriminator Backpropagates to G	No Gradient Backpropagation from Discriminator
Input Type	Noise Vector	Masked Text Inputs
Use of Noise Vector	Yes	No

EXPERIEMENT

Evaluation Datasets:

- **GLUE(General Language Understanding Evaluation) Score:**
  - Tasks: Textual Entailment (RTE, MNLI), Question-Answer Entailment (QNLI), Paraphrase (MRPC, QQP), Textual Similarity (STS), Sentiment (SST), Linguistic Acceptability (CoLA).

GLUE Metrics:

- Accuracy (Other tasks)
- Results reported as average score over all tasks.

- **SQuAD(Stanford Question Answering) Dataset:**
  - Tasks: Versions: 1.1 (Span-based QA)
  - 2.0 (Unanswerable questions).

SQuAD Metrics: similar to Exact Match (EM) and F1 Score.

Pre-training Data:

- **Small Model:** Wikipedia & BooksCorpus (3.3 Billion tokens) [Same as BERT].
- **Large Model:** XLNet Data (33 Billion tokens) – Wikipedia, BooksCorpus, ClueWeb, CommonCrawl, Gigaword.

Fine-Tuning:

- GLUE Fine-Tuning: Adds simple linear classifiers on top of ELECTRA.
- SQuAD Fine-Tuning: Uses XLNet's QA module

Evaluation Approach:

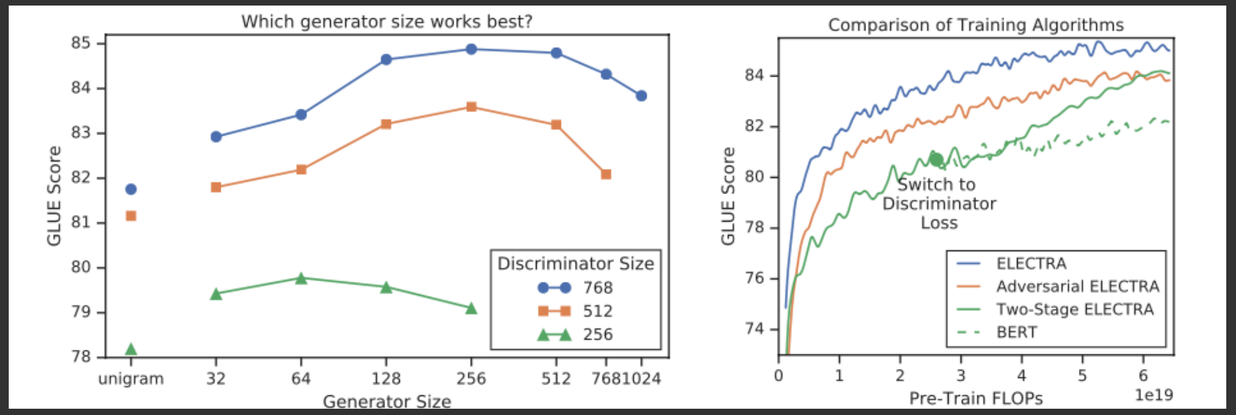
- Fine-tuned results are median of 10 runs from the same pre-trained checkpoint to reduce randomness.
- All evaluations are on **English data.**

EXPERIEMENTS

Weight Sharing Problem:

- If the generator and discriminator are the same size, all of the transformer weights can be tied.
- For generator and discriminator are the same size comparison of Weight Tying Strategies:
  - No weight tying: GLUE score = 83.6.
  - Tying token embeddings: GLUE score = 84.3.
  - Tying all weights: GLUE score = 84.4.
- Smaller generator is more efficient because only embeddings (both token and positional embeddings) are shared.
- Benefits of Tied Token Embeddings(small generator):
  - Masked language modeling effectively learns token representations.
  - Generator's softmax densely updates all token embeddings.
- Result:
  - Use tied embeddings for further experiments but not for same size of generator and discriminator.

EXPERIEMENTS



Small generators:

- Smaller generators reduce computational costs.
- Using equal-sized generator and discriminator doubles compute per step.
- Models are made smaller by reducing layer sizes, keeping other settings constant.
- in this fig All models are trained for 500k steps.
- Best performance is achieved when the generator is 1/4 to 1/2 the size of the discriminator.
- Too strong a generator makes learning harder for the discriminator.
- The discriminator may focus on the generator instead of the actual data.

Alternative Training Algorithms: (Tried but not improved results)

- Two-Stage Training: The generator is first trained with LMLM for n steps.
- The discriminator is then initialized with the generator's weights and trained with LDisc for n steps.
- Requires equal-sized generator and discriminator for proper weight initialization.
- Without initialization, the discriminator fails to learn beyond the majority class.
- Joint training works better as the generator starts weak and improves gradually.
- Adversarial Training: Faced two key problems.

1. Lower accuracy in masked language modeling (58% vs. 65% with MLE).
2. The generator produced low-entropy output, reducing token diversity.

EXPERIEMENTS

Comparison of Small models perfomance

Model	Train / Infer FLOPs	Speedup	Params	Train Time + Hardware	GLUE
ELMo	3.3e18 / 2.6e10	19x / 1.2x	96M	14d on 3 GTX 1080 GPUs	71.2
GPT	4.0e19 / 3.0e10	1.6x / 0.97x	117M	25d on 8 P6000 GPUs	78.8
BERT-Small	1.4e18 / 3.7e9	45x / 8x	14M	4d on 1 V100 GPU	75.1
BERT-Base	6.4e19 / 2.9e10	1x / 1x	110M	4d on 16 TPUv3s	82.2
ELECTRA-Small	1.4e18 / 3.7e9	45x / 8x	14M	4d on 1 V100 GPU	79.9
50% trained	7.1e17 / 3.7e9	90x / 8x	14M	2d on 1 V100 GPU	79.0
25% trained	3.6e17 / 3.7e9	181x / 8x	14M	1d on 1 V100 GPU	77.7
12.5% trained	1.8e17 / 3.7e9	361x / 8x	14M	12h on 1 V100 GPU	76.0
6.25% trained	8.9e16 / 3.7e9	722x / 8x	14M	6h on 1 V100 GPU	74.1
ELECTRA-Base	6.4e19 / 2.9e10	1x / 1x	110M	4d on 16 TPUv3s	85.1

Comparison of small models on the GLUE dev set

ELECTRA-Small: (Key Changes compare to BERT-Base)

- Designed for efficiency and trained on a single GPU.
- Reduced sequence length (512 → 128).
- Reduced batch size (256 → 128).
- Reduced hidden dimension size (768 → 256).
- Reduced token embedding size (768 → 128).
- Trained for 1M steps to match the same training FLOPs(floating point operations - arithmetic operations) as BERT-Small (1.5M steps).

Results:

- Achieves higher GLUE score compared to BERT-Small, despite using the same compute.
- Reasonable performance even when trained for as little as 6 hours.
- Outperforms GPT, even with significantly fewer resources.

EXPERIEMENTS

Comparison of large models perfomance

Model	Train FLOPs	Params	CoLA	SST	MRPC	STS	QQP	MNLI	QNLI	RTE	Avg.
BERT	1.9e20 (0.27x)	335M	60.6	93.2	88.0	90.0	91.3	86.6	92.3	70.4	84.0
RoBERTa-100K	6.4e20 (0.90x)	356M	66.1	95.6	**91.4**	92.2	92.0	89.3	94.0	82.7	87.9
RoBERTa-500K	3.2e21 (4.5x)	356M	68.0	96.4	90.9	92.1	92.2	90.2	94.7	86.6	88.9
XLNet	3.9e21 (5.4x)	360M	69.0	**97.0**	90.8	92.2	92.3	90.8	94.9	85.9	89.1
BERT (ours)	7.1e20 (1x)	335M	67.0	95.9	89.1	91.2	91.5	89.6	93.5	79.5	87.2
ELECTRA-400K	7.1e20 (1x)	335M	**69.3**	96.0	90.6	92.1	**92.4**	90.5	94.5	86.8	89.0
ELECTRA-1.75M	3.1e21 (4.4x)	335M	69.1	96.9	90.8	**92.6**	**92.4**	**90.9**	**95.0**	**88.0**	**89.5**

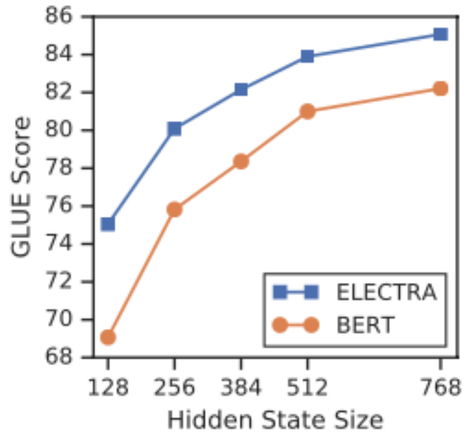
Comparison of large models on the GLUE dev set

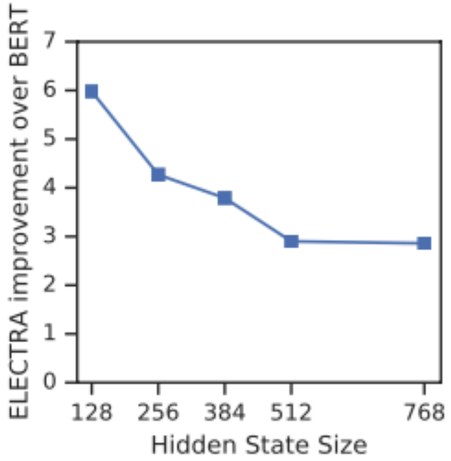
Model	Train FLOPs	CoLA	SST	MRPC	STS	QQP	MNLI	QNLI	RTE	WNLI	Avg.*	Score
BERT	1.9e20 (0.06x)	60.5	94.9	85.4	86.5	89.3	86.7	92.7	70.1	65.1	79.8	80.5
RoBERTa	3.2e21 (1.02x)	67.8	96.7	89.8	91.9	90.2	90.8	95.4	88.2	89.0	88.1	88.1
ALBERT	3.1e22 (10x)	69.1	**97.1**	**91.2**	92.0	90.5	**91.3**	–	89.2	91.8	89.0	–
XLNet	3.9e21 (1.26x)	70.2	**97.1**	90.5	**92.6**	90.4	90.9	–	88.5	**92.5**	89.1	–
ELECTRA	3.1e21 (1x)	**71.7**	**97.1**	90.7	92.5	**90.8**	**91.3**	**95.8**	**89.8**	**92.5**	**89.5**	**89.4**

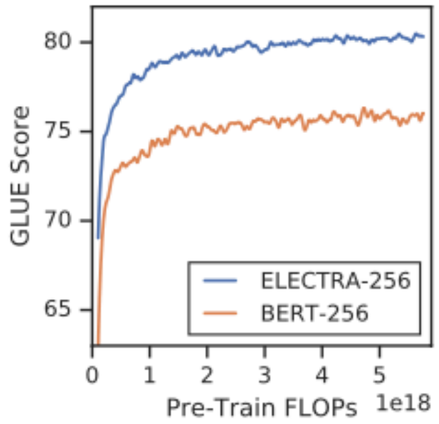
GLUE test-set results for large models

RESULTS:

Model	ELECTRA	All-Tokens MLM	Replace MLM	ELECTRA 15%	BERT
GLUE score	85.0	84.3	82.4	82.4	82.2







CONCLUSION

- **High Efficiency:** Learns from all input tokens using Replaced Token Detection (RTD), unlike BERT's masked token approach.
- **Faster Training:** Requires fewer training steps and with fewer resources performance.
- **Better Performance:** Outperforms BERT on NLP benchmarks like GLUE, SQuAD and even with smaller models.
- **Resource Efficiency:** Smaller models (e.g., ELECTRA-Small) achieve performance comparable to much larger models (e.g., BERT-Base).

Application of ELECTRA:

- Sentiment Analysis
- Named Entity Recognition (NER)
- Question Answering Systems
- Text Classification

QUESTIONS?