# CodeBERT: A Pre-trained Model for Programming and Natural Languages

Duyu Tang
Natural Language Computing Group
Microsoft Research Asia

# CodeBERT is on arXiv

https://arxiv.org/pdf/2002.08155



## CodeBERT: A Pre-Trained Model for Programming and Natural Languages

Zhangyin Feng [*1] Daya Guo [*2] Duyu Tang [3] Nan Duan [3] Xiaocheng Feng [1] Ming Gong [4] Linjun Shou [4]
Bing Qin [1] Ting Liu [1] Daxin Jiang [4] Ming Zhou [3]

### Abstract

We present CodeBERT, a *bimodal* pre-trained model for programming language (PL) and natural language (NL). CodeBERT learns general-purpose representations that support downstream NL-PL applications such as natural language code search, code documentation generation, etc. We develop CodeBERT with Transformer-based neural architecture, and train it with a hybrid objective function that incorporates the pre-training task of replaced token detection, which is to detect plausible alternatives sampled from generators. This enables us to utilize both "*bimodal*" data of NL-PL pairs and "*unimodal*" data, where the former provides input tokens for model training while the latter helps to learn better generators. We evaluate CodeBERT on two NL-PL applications by fine-tuning model parameters. Results show that CodeBERT achieves state-of-the-art performance on both natural language code search and code documentation generation. Furthermore, to investigate what type of knowledge is learned in CodeBERT, we construct a dataset for NL-PL probing, and evaluate in a zero-shot setting where parameters of pre-trained models are fixed. Results show that CodeBERT performs better than previous pre-trained models on NL-PL probing.

### 1. Introduction

Large pre-trained models such as ELMo (Peters et al., 2018), GPT (Radford et al., 2018), BERT (Devlin et al., 2018), XLNet (Yang et al., 2019) and RoBERTa (Liu et al., 2019) have dramatically improved the state-of-the-art on a variety of natural language processing (NLP) tasks. These pre-trained models learn effective contextual representations from massive unlabeled text optimized by self-supervised objectives, such as masked language modeling, which predicts the original masked word from an artificially masked input sequence. The success of pre-trained models in NLP also drives a surge of multi-modal pre-trained models, such as ViLBERT (Lu et al., 2019) for language-image and VideoBERT (Sun et al., 2019) for language-video, which are learned from *bimodal* data such as language-image pairs with *bimodal* self-supervised objectives.

In this work, we present CodeBERT, a *bimodal* pre-trained model for natural language (NL) and programming language (PL) like Python, Java, JavaScript, etc. CodeBERT captures the semantic connection between natural language and programming language, and produces general-purpose representations that can broadly support NL-PL understanding tasks (e.g. natural language code search) and generation tasks (e.g. code documentation generation). It is developed with the multi-layer Transformer (Vaswani et al., 2017), which is adopted in a majority of large pre-trained models. In order to make use of both *bimodal* instances of NL-PL pairs and large amount of available *unimodal* codes, we train CodeBERT with a hybrid objective function, including standard masked language modeling (Devlin et al., 2018) and replaced token detection (Clark et al., 2020), where *unimodal* codes help to learn better generators for producing better alternative tokens for the latter objective.

We train CodeBERT from Github code repositories in 6 programming languages, where *bimodal* datapoints are codes that pair with function-level natural language documentations (Husain et al., 2019). Training is conducted in a setting similar to that of multilingual BERT (Pires et al., 2019), in which case one pre-trained model is learned for 6 programming languages with no explicit markers used to denote the input programming language. We evaluate CodeBERT on two downstream NL-PL tasks, including natural language code search and code documentation generation. Results show that fine-tuning the parameters of CodeBERT achieves state-of-the-art performance on both tasks. To further investigate what type of knowledge is learned in CodeBERT, we construct a dataset for NL-PL probing, and test CodeBERT in a zero-shot scenario, i.e. without fine-tuning the parameters of CodeBERT. We find that CodeBERT consistently outperforms RoBERTa, a purely natural language-based pre-trained model.

*Work is done during internship at Microsoft Research Asia. Contact: dutang@microsoft.com [1]Harbin Institute of Technology, Harbin,China [2]Sun Yat-sen University, China [3]Microsoft Research Asia, Beijing, China [4]Microsoft Search Technology Center Asia, Beijing, China.

Google Scholar

TITLE

BERT: Pre-tr
J Devlin, MW C
https://arxiv.org

Load forecas
BJ Chen, MW
Power Systems

Semantic Pa
Base
W Yih, MW Cha
ACL

Guiding sem
MW Chang, L F
Proceedings of

# BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

| | |
|---|---|
| Authors | Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova |
| Publication date | 2018/10/11 |
| Journal | https://arxiv.org/abs/1810.04805 |
| Description | We introduce a new language representation model called BERT, which stands for Bidirectional Encoder Representations from Transformers. Unlike recent language representation models, BERT is designed to pre-train deep bidirectional representations from unlabeled text by jointly conditioning on both left and right context in all layers. As a result, 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 task-specific architecture modifications. |

ptually simple and empirically powerful. It obtains new state-of-the-art
natural language processing tasks, including pushing the GLUE score
absolute improvement), MultiNLI accuracy to 86.7%(4.6% absolute
v1. 1 question answering Test F1 to 93.2 (1.5 point absolute
AD v2. 0 Test F1 to 83.1 (5.1 point absolute improvement).

Total c

3414

2018 2019 2020

## SQuAD benchmark

SQuAD1.1 Leaderboard

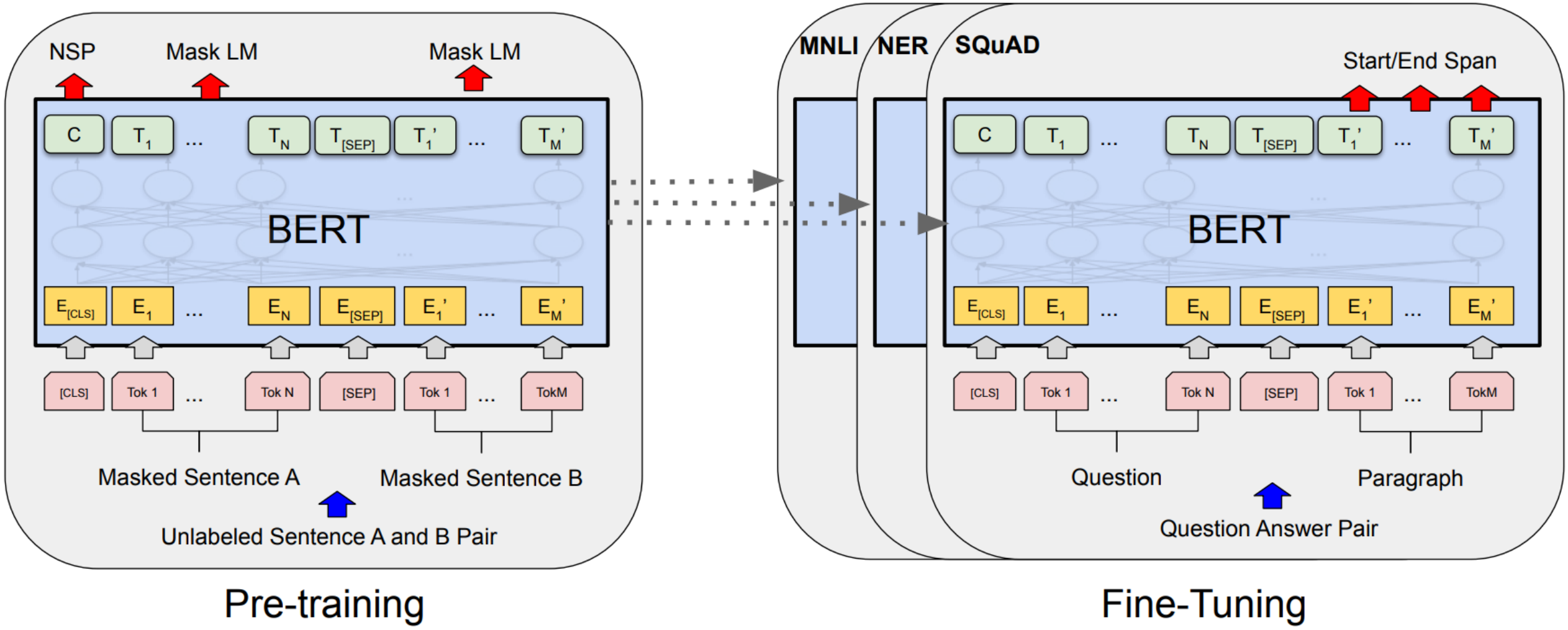
| Rank | Model | EM | F1 |
|---|---|---|---|
| | Human Performance<br>*Stanford University*<br>(Rajpurkar et al. '16) | 82.304 | 91.221 |
| 1<br>Oct 05, 2018 | BERT (ensemble)<br>*Google AI Language*<br>https://arxiv.org/abs/1810.04805 | **87.433** | **93.160** |
| 2<br>Sep 09, 2018 | nlnet (ensemble)<br>*Microsoft Research Asia* | 85.356 | 91.202 |
| 3<br>Jul 11, 2018 | QANet (ensemble)<br>*Google Brain & CMU* | 84.454 | 90.490 |

## GLUE benchmark

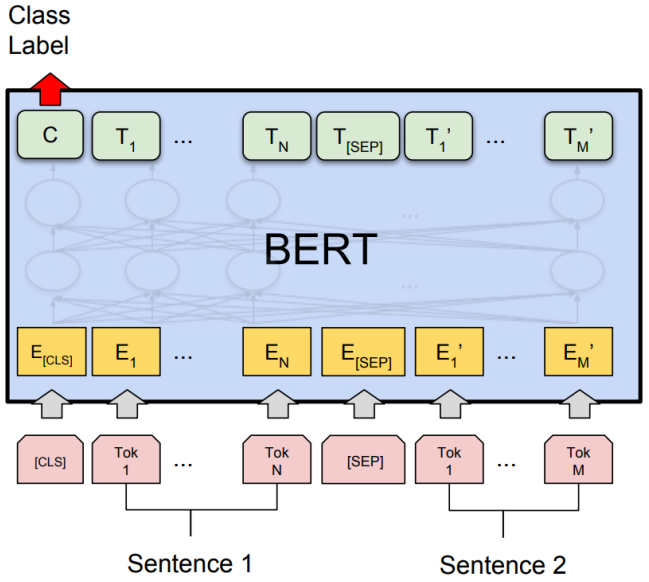
| Rank | Model | Score | CoLA | SST-2 | MRPC | STS-B | QQP | MNLI-m | QNLI | RTE |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | BERT: 24-layers, 1024-hidden, 16-heads | 80.4 | 60.5 | 94.9 | 85.4/89.3 | 87.6/86.5 | 89.3/72.1 | 86.7 | 91.1 | 70.1 |
| 2 | Singletask Pretrain Transformer | 72.8 | 45.4 | 91.3 | 75.7/82.3 | 82.0/80.0 | 88.5/70.3 | 82.1 | 88.1 | 56.0 |
| 3 | BiLSTM+ELMo+Attn | 70.5 | 36.0 | 90.4 | 77.9/84.9 | 75.1/73.3 | 84.7/64.8 | 76.4 | 79.9 | 56.8 |

Source: https://ai.googleblog.com/2018/11/open-sourcing-bert-state-of-art-pre.html

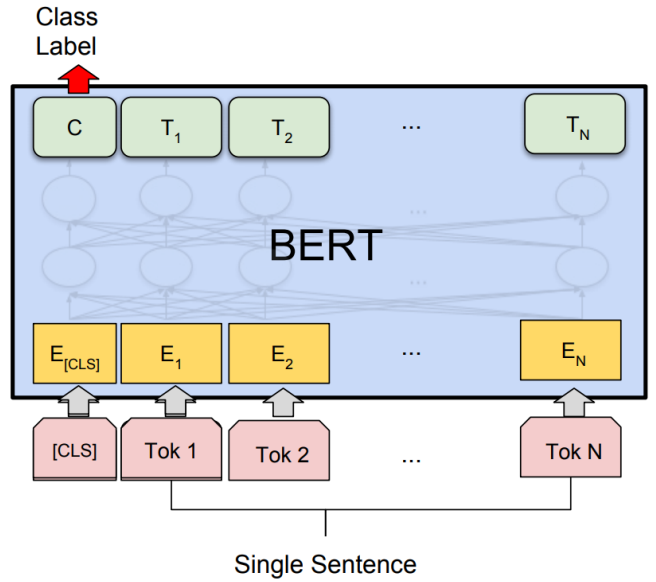
# Pre-training and fine-tuning procedures for BERT



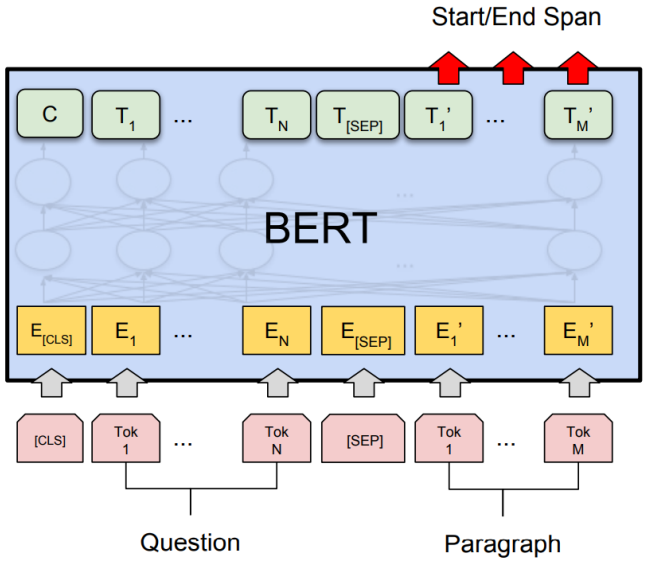
# Fine-tuning BERT on Different Tasks



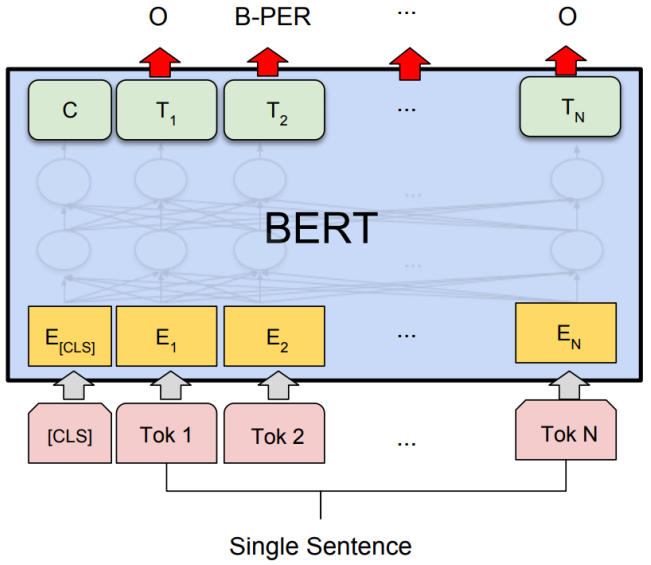

(a) Sentence Pair Classification Tasks: MNLI, QQP, QNLI, STS-B, MRPC, RTE, SWAG




(b) Single Sentence Classification Tasks: SST-2, CoLA

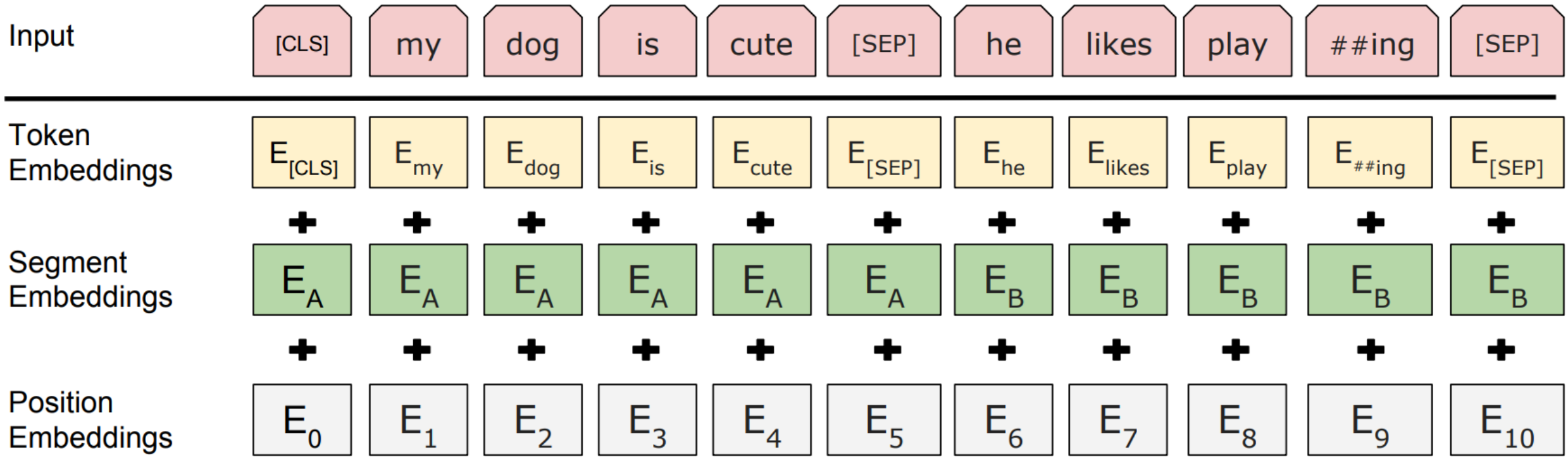



(c) Question Answering Tasks: SQuAD v1.1

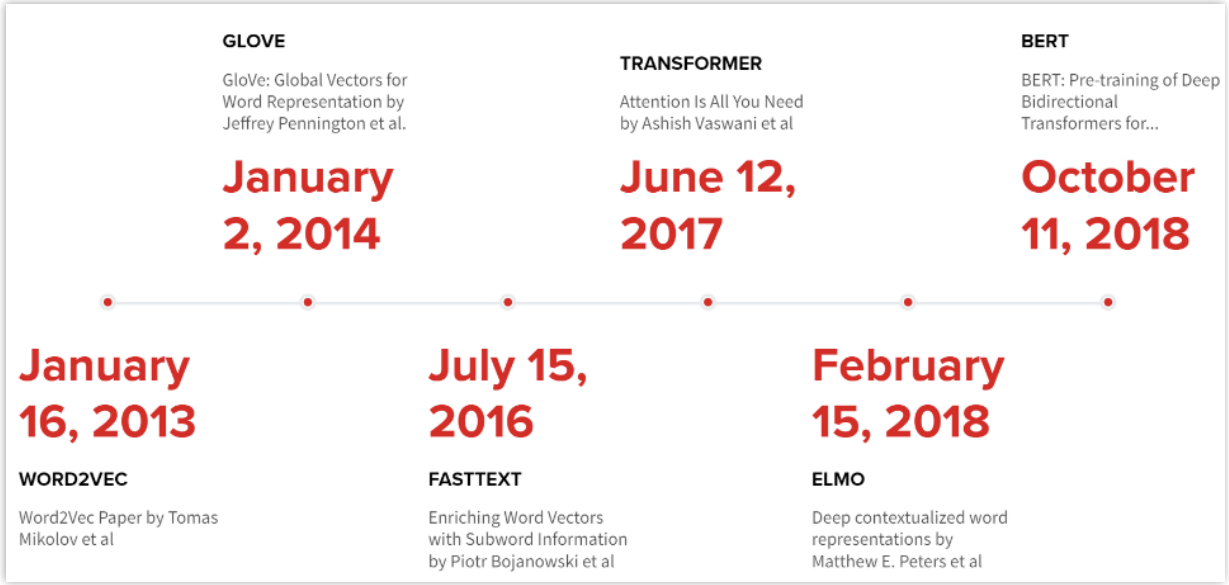


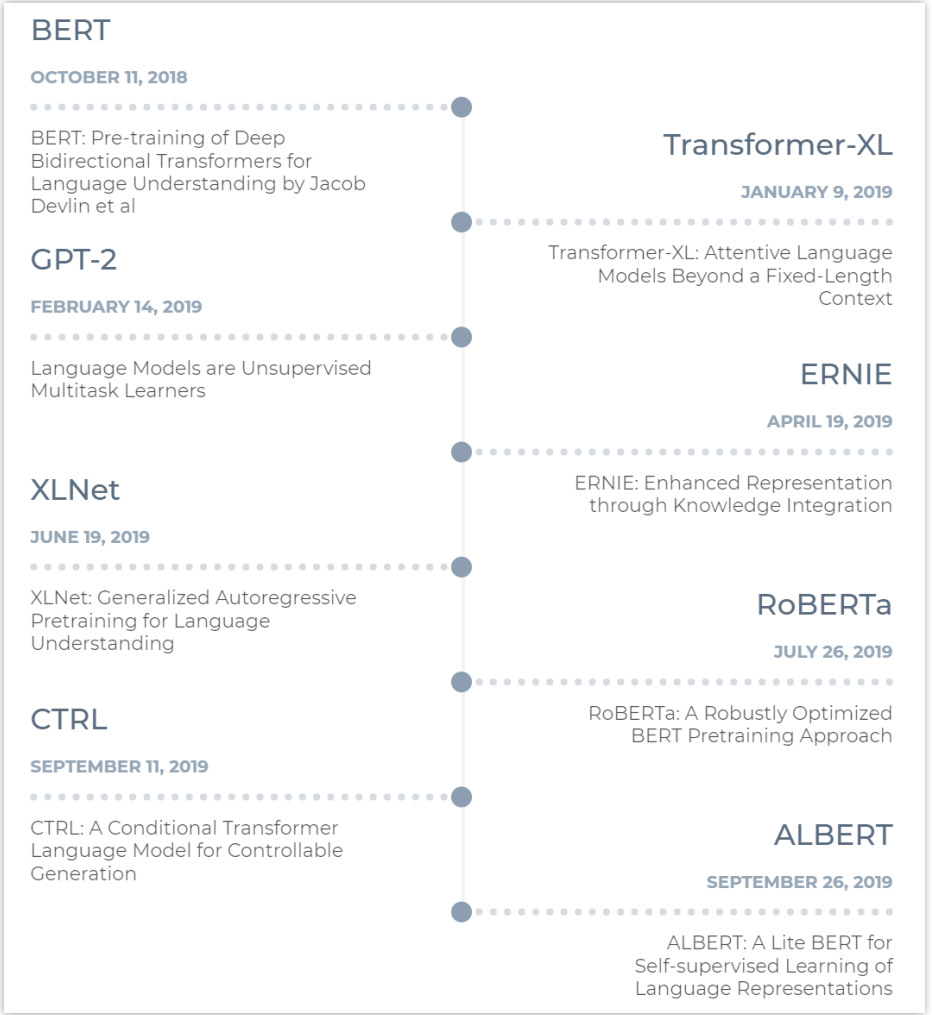

(d) Single Sentence Tagging Tasks: CoNLL-2003 NER

# BERT input representation



# Timeline of some major projects before and after BERT




Source: https://towardsdatascience.com/2019-year-of-bert-and-transformer-f200b53d05b9

arXiv.org > cs > arXiv:1908.02265

Computer Science > Computer Vision and Pattern Recognition

## ViLBERT: Pretraining Task-Agnostic Visiolinguistic Representations for Vision-and-Language Tasks

Jiasen Lu, Dhruv Batra, Devi Parikh, Stefan Lee

*(Submitted on 6 Aug 2019)*

arXiv.org > cs > arXiv:1908.07490

Computer Science > Computation and Language

## LXMERT: Learning Cross-Modality Encoder Representations from Transformers

Hao Tan, Mohit Bansal

*(Submitted on 20 Aug 2019 (v1), last revised 3 Dec 2019 (this version, v3))*

arXiv.org > cs > arXiv:1904.01766

Computer Science > Computer Vision and Pattern Recognition

## VideoBERT: A Joint Model for Video and Language Representation Learning

Chen Sun, Austin Myers, Carl Vondrick, Kevin Murphy, Cordelia Schmid

*(Submitted on 3 Apr 2019 (v1), last revised 11 Sep 2019 (this version, v2))*

arXiv.org > cs > arXiv:1908.03557

Computer Science > Computer Vision and Pattern Recognition

## VisualBERT: A Simple and Performant Baseline for Vision and Language

Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, Kai-Wei Chang

*(Submitted on 9 Aug 2019)*

arXiv.org > cs > arXiv:1908.08530

Computer Science > Computer Vision and Pattern Recognition

## VL-BERT: Pre-training of Generic Visual-Linguistic Representations

Weijie Su, Xizhou Zhu, Yue Cao, Bin Li, Lewei Lu, Furu Wei, Jifeng Dai

*(Submitted on 22 Aug 2019 (v1), last revised 18 Feb 2020 (this version, v4))*

arXiv.org > cs > arXiv:1908.06066

Computer Science > Computer Vision and Pattern Recognition

## Unicoder-VL: A Universal Encoder for Vision and Language by Cross-modal Pre-training

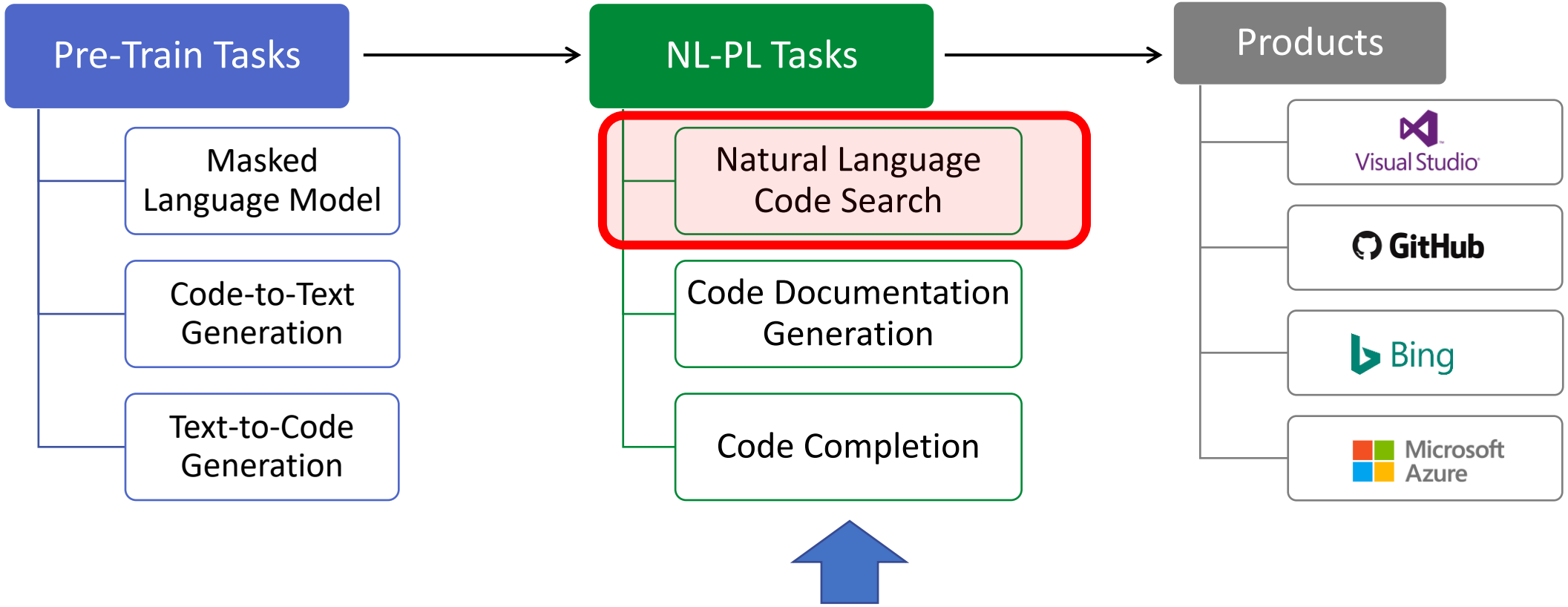
Gen Li, Nan Duan, Yuejian Fang, Ming Gong, Daxin Jiang, Ming Zhou

*(Submitted on 16 Aug 2019 (v1), last revised 2 Dec 2019 (this version, v3))*

# From Unimodal to Bimodal

As of Sep 2019.

# Landscape of CodeBERT

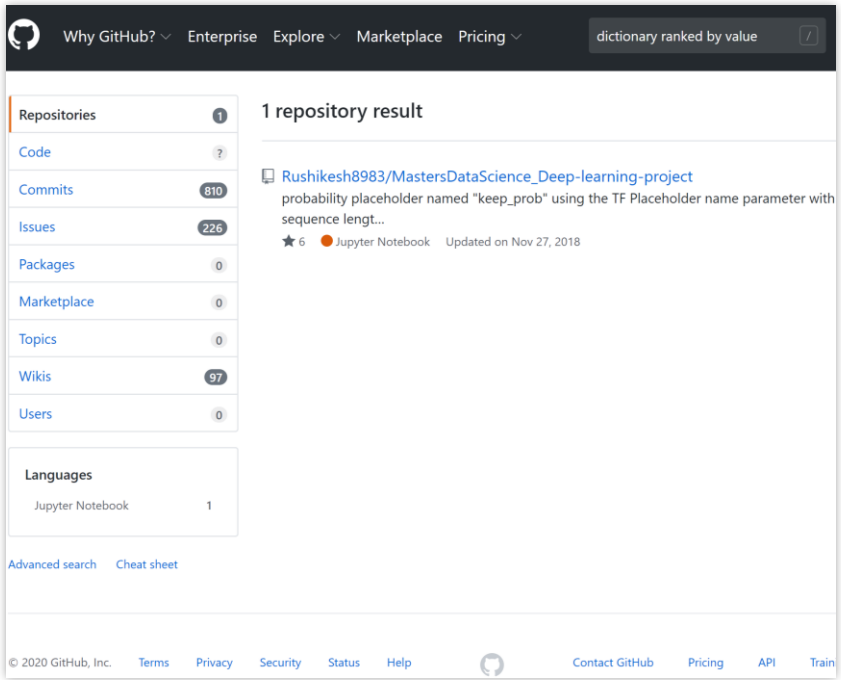


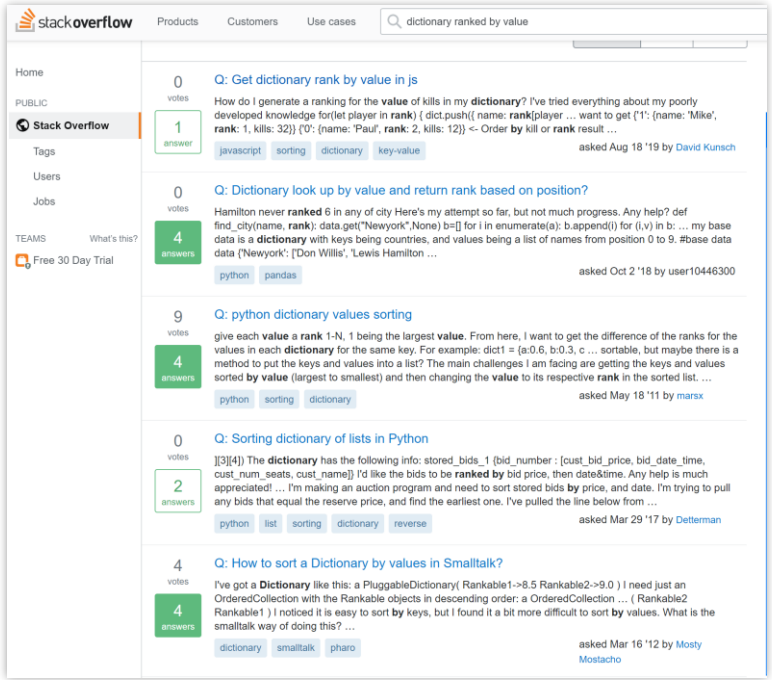

**CodeBERT**

A pre-trained model for natural language (NL) and programming language (PL)

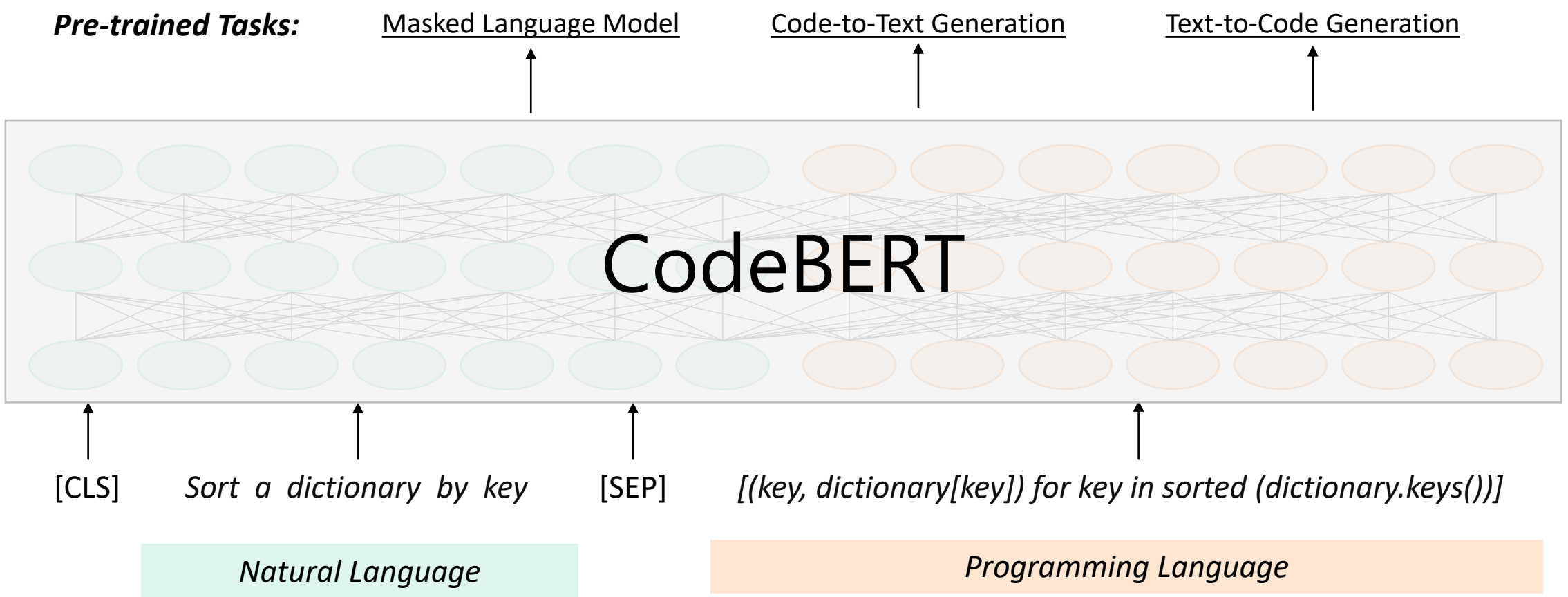
# Challenge: Semantic Matching between NL and PL

**query** = "*dictionary ranked by value*"

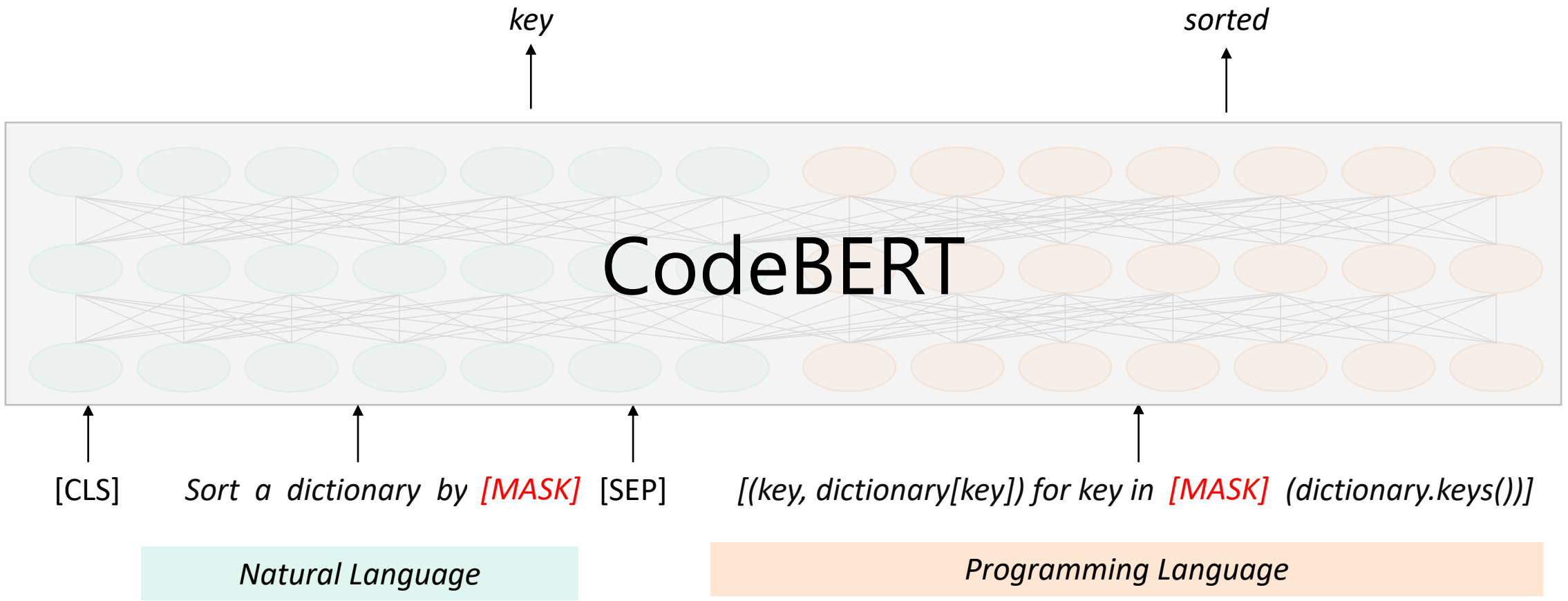


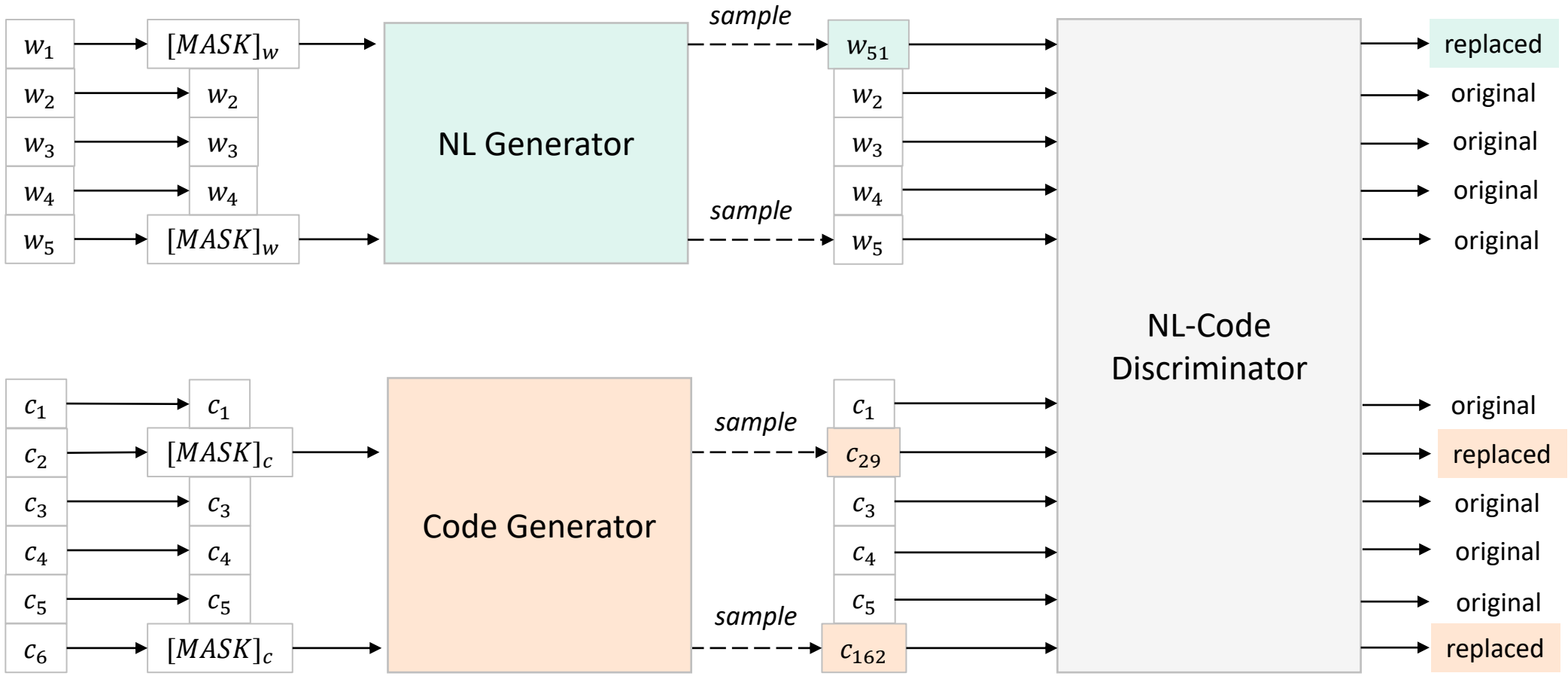
# CodeBERT: An NL-PL Pre-trained Model



# Learning Objective #1: Masked Language Model



# Learning Objective #2: Replaced Token Detection



# Training Data

## NL-PL pairs from CodeSearchNet Challenge

### Training Data

| | Number of Functions | |
|---|---|---|
| | w/ documentation | All |
| Go | 347 789 | 726 768 |
| Java | 542 991 | 1 569 889 |
| JavaScript | 157 988 | 1 857 835 |
| PHP | 717 313 | 977 821 |
| Python | 503 502 | 1 156 085 |
| Ruby | 57 393 | 164 048 |
| All | 2 326 976 | 6 452 446 |

### Model Parameters

| | |
|---|---|
| Number of Layers | 12 |
| Hidden size | 768 |
| FFN inner hidden size | 3072 |
| Attention heads | 12 |
| Attention head size | 64 |
| Total number of parameters | 124,696,665 |

**CodeSearchNet Challenge: Evaluating the State of Semantic Code Search**. Hamel Husain, Ho-Hsiang Wu, Tiferet Gazit, Miltiadis Allamanis, Marc Brockschmidt. Arxiv-2019

# A Training Example

```
def _parse_memory(s):
    """
    Parse a memory string in the format supported by Java (e.g. 1g, 200m) and
    return the value in MiB

    >>> _parse_memory("256m")
    256
    >>> _parse_memory("2g")
    2048
    """
    units = {'g': 1024, 'm': 1, 't': 1 << 20, 'k': 1.0 / 1024}
    if s[-1].lower() not in units:
        raise ValueError("invalid format: " + s)
    return int(float(s[:-1]) * units[s[-1].lower()])
```

**Natural language**: first paragraph of documentation

**Code:** function

Source: https://github.com/apache/spark/blob/618d6bff71073c8c93501ab7392c3cc579730f0b/python/pyspark/rdd.py#L125-L138

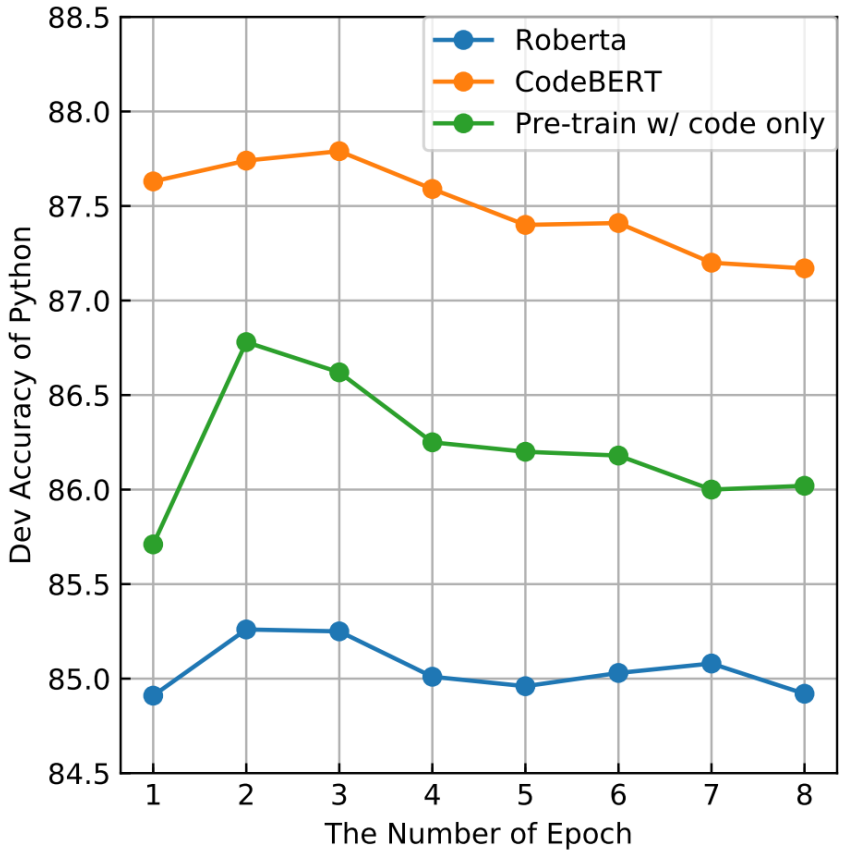
# Application 1: Natural Language Code Search

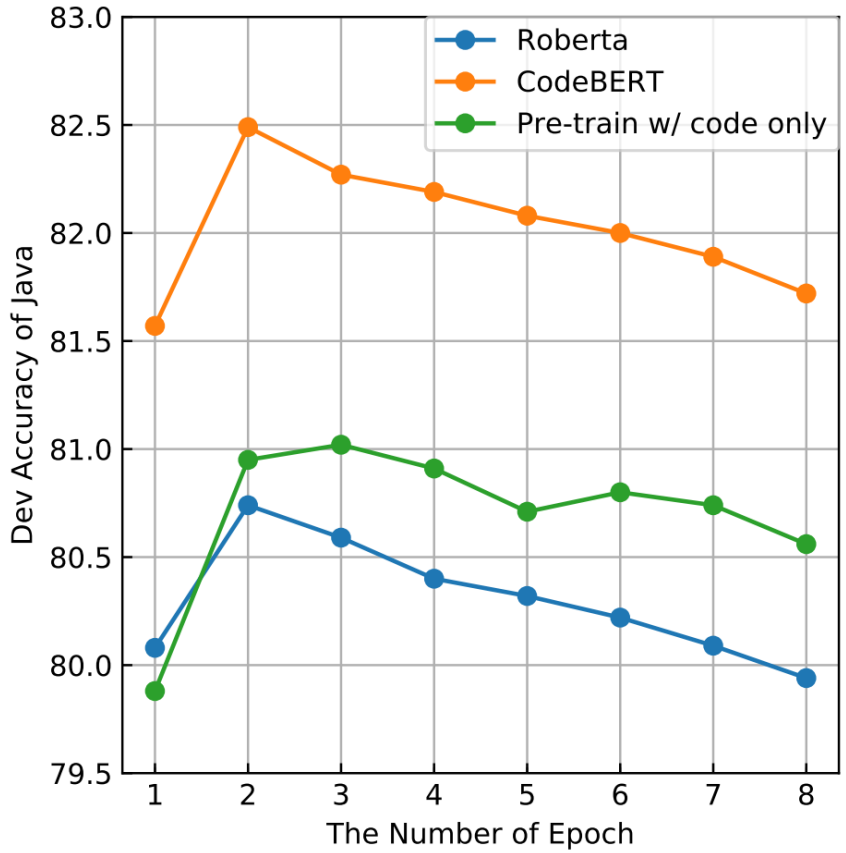
- State-of-the-art on CodeSearchNet Corpus
  - Fine-tune on 6 programming languages (Ruby, JavaScript, Go, Python, Java, PHP)

| MODEL | RUBY | JAVASCRIPT | GO | PYTHON | JAVA | PHP | MA-AVG |
|---|---|---|---|---|---|---|---|
| NBOW | 0.4285 | 0.4607 | 0.6409 | 0.5809 | 0.5140 | 0.4835 | 0.5181 |
| CNN | 0.2450 | 0.3523 | 0.6274 | 0.5708 | 0.5270 | 0.5294 | 0.4753 |
| BIRNN | 0.0835 | 0.1530 | 0.4524 | 0.3213 | 0.2865 | 0.2512 | 0.2580 |
| SELFATT | 0.3651 | 0.4506 | 0.6809 | 0.6922 | 0.5866 | 0.6011 | 0.5628 |
| ROBERTA | 0.6245 | 0.6060 | 0.8204 | 0.8087 | 0.6659 | 0.6576 | 0.6972 |
| PT W/ CODE ONLY (INIT=SCRATCH) | 0.5712 | 0.5557 | 0.7929 | 0.7855 | 0.6567 | 0.6172 | 0.6632 |
| PT W/ CODE ONLY (INIT=ROBERTA) | 0.6612 | 0.6402 | 0.8191 | 0.8438 | 0.7213 | 0.6706 | 0.7260 |
| CODEBERT (MLM, INIT=SCRATCH) | 0.5695 | 0.6029 | 0.8304 | 0.8261 | 0.7142 | 0.6556 | 0.6998 |
| CODEBERT (MLM, INIT=ROBERTA) | 0.6898 | 0.6997 | 0.8383 | 0.8647 | 0.7476 | 0.6893 | 0.7549 |
| CODEBERT (RTD, INIT=ROBERTA) | 0.6414 | 0.6512 | 0.8285 | 0.8263 | 0.7150 | 0.6774 | 0.7233 |
| CODEBERT (MLM+RTD, INIT=ROBERTA) | **0.6926** | **0.7059** | **0.8400** | **0.8685** | **0.7484** | **0.7062** | **0.7603** |

Training 1,000 batches of data costs 600 minutes with MLM objective, 120 minutes with RTD objective.

# Learning curve in fine-tuning procedure



# Application 2: Code-to-Text Generation

- State-of-the-art on CodeSearchNet Corpus
- Fine-tune on 6 programming languages
  - CodeBERT as the encoder
  - Transformer as decoder (6 layer, 768d hidden states and 12 attention heads)

| MODEL | RUBY | JAVASCRIPT | GO | PYTHON | JAVA | PHP | OVERALL |
|---|---|---|---|---|---|---|---|
| SEQ2SEQ | 6.96 | 6.88 | 23.48 | 13.04 | 11.42 | 18.40 | 13.36 |
| TRANSFORMER | 7.87 | 8.14 | 25.61 | 13.44 | 12.57 | 18.25 | 14.31 |
| ROBERTA | 7.26 | 5.72 | 26.09 | 14.92 | 13.20 | 19.90 | 14.52 |
| PRE-TRAIN W/ CODE ONLY | 7.39 | 8.30 | 26.39 | 15.05 | 13.07 | 20.71 | 15.15 |
| CODEBERT (RTD) | 7.36 | 8.73 | 26.02 | 15.12 | 12.72 | 20.25 | 15.03 |
| CODEBERT (MLM) | 7.95 | 8.51 | **26.79** | **15.48** | 13.59 | 21.00 | 15.55 |
| CODEBERT (MLM+RTD) | **8.46** | **9.54** | 26.66 | 15.41 | **14.56** | **21.32** | **15.99** |

# Generalization to programming languages NOT in pre-training

| Model | BLEU |
|---|---|
| MOSES (Koehn et al., 2007) | 11.57 |
| IR | 13.66 |
| SUM-NN (Rush et al., 2015) | 19.31 |
| 2-layer BiLSTM | 19.78 |
| Transformer (Vaswani et al., 2017) | 19.68 |
| TreeLSTM (Tai et al., 2015) | 20.11 |
| CodeNN (Iyer et al., 2016) | 20.53 |
| code2seq (Alon et al., 2019) | **23.04** |
| RoBERTa | 19.81 |
| pre-train w/ code only | 20.65 |
| CodeBERT (RTD) | 22.14 |
| CodeBERT (MLM) | 22.32 |
| CodeBERT (MLM+RTD) | **22.36** |

- 66,015 pairs of questions and answers (C# codes) automatically collected from StackOverflow

- Encoder: CodeBERT
- Decoder: 2-layer GRU

Iyer, S., Konstas, I., Cheung, A., and Zettlemoyer, L. **Summarizing source code using a neural attention model**. ACL-2016

# Application 3: Probing

- Predict masked tokens with CodeBERT, without fine-tuning

masked NL token

*"Transforms a vector np.arange(-N, M, dx) to np.arange( min (|vec|), max(N,M),dx)]"*

```
def vec_to_halfvec(vec):

    d = vec[1:] - vec[:-1]
    if ((d/d.mean()).std() > 1e-14) or (d.mean() < 0):
        raise ValueError('vec must be np.arange() in increasing order')
    dx = d.mean()
    lowest = np.abs(vec).min()
    highest = np.abs(vec).max()
    return np.arange(lowest, highest + 0.1*dx, dx).astype(vec.dtype)
```

masked PL token

| | | *max* | *min* | *less* | *greater* |
|---|---|---|---|---|---|
| NL | Roberta | 96.24% | 3.73% | 0.02% | 0.01% |
| | CodeBERT-MLM | 39.38% | 60.60% | 0.02% | 0.0003% |
| PL | Roberta | 95.85% | 4.15% | - | - |
| | CodeBERT-MLM | 0.001% | 99.999% | - | - |

# Probing Results

| | RUBY | JAVASCRIPT | GO | PYTHON | JAVA | PHP | ALL |
|---|---|---|---|---|---|---|---|
| **NUMBER OF DATAPOINTS FOR PROBING** | | | | | | | |
| PL (2 CHOICES) | 38 | 272 | 152 | 1,264 | 482 | 407 | 2,615 |
| NL (4 CHOICES) | 20 | 65 | 159 | 216 | 323 | 73 | 856 |
| **PL PROBING** | | | | | | | |
| ROBERTA | 73.68% | 65.07% | 71.05% | 59.02% | 62.03% | 70.02% | 62.83% |
| PRE-TRAIN W/ CODE ONLY | **84.21%** | 81.62% | 91.45% | 75.16% | 85.27% | 83.05% | 80.00% |
| CODEBERT (MLM) | 81.58% | **86.40%** | **92.11%** | **79.19%** | **91.08%** | **90.42%** | **84.67%** |
| **PL PROBING WITH PRECEDING CONTEXT ONLY** | | | | | | | |
| ROBERTA | **71.05%** | 51.84% | 51.32% | 55.06% | 42.12% | 52.58% | 51.97% |
| PRE-TRAIN W/ CODE ONLY | 63.16% | 49.26% | **59.51%** | 56.96% | **59.13%** | 58.72% | 57.05% |
| CODEBERT (MLM) | 60.53% | **52.21%** | **59.51%** | **61.16%** | 57.68% | **61.92%** | **59.58%** |
| **NL PROBING** | | | | | | | |
| ROBERTA | 45.00% | 72.31% | 47.17% | 67.59% | 50.77% | 61.64% | 56.78% |
| PRE-TRAIN W/ CODE ONLY | 55.00% | 61.54% | 55.97% | 65.74% | 53.25% | 61.64% | 58.29% |
| CODEBERT (MLM) | **60.00%** | **83.08%** | **64.15%** | **72.69%** | **61.61%** | **75.34%** | **67.64%** |

# Takeaways

- CodeBERT, a large pre-trained model for multiple programming langs
- Show strong performance on code search, code-to-text generation
- Introduce a new NL-PL probing task

# Thanks!

Contact: dutang@microsoft.com