

# 有监督学习和无监督学习

---

这是我看吴恩达视频截取的两张ppt，以跟self-supervised learning对照学习。

**supervised learning**是从有标签的数据中去学习，以便将来可以根据数据predict标签。

## Supervised learning

Learns from being given “**right answers**”

### Regression

Predict a **number**

**infinitely** many possible outputs

### Classification

predict **categories**

**small number of possible outputs**

**unsupervised learning**不需要数据的标签，它的目的是去自动寻找数据中的结构、特征或模式。

## Unsupervised learning

Data only comes with inputs  $x$ , but not output labels  $y$ .

Algorithm has to find **structure** in the data.

### Clustering

Group similar data points together.

### Dimensionality reduction

Compress data using fewer numbers.

### Anomaly detection

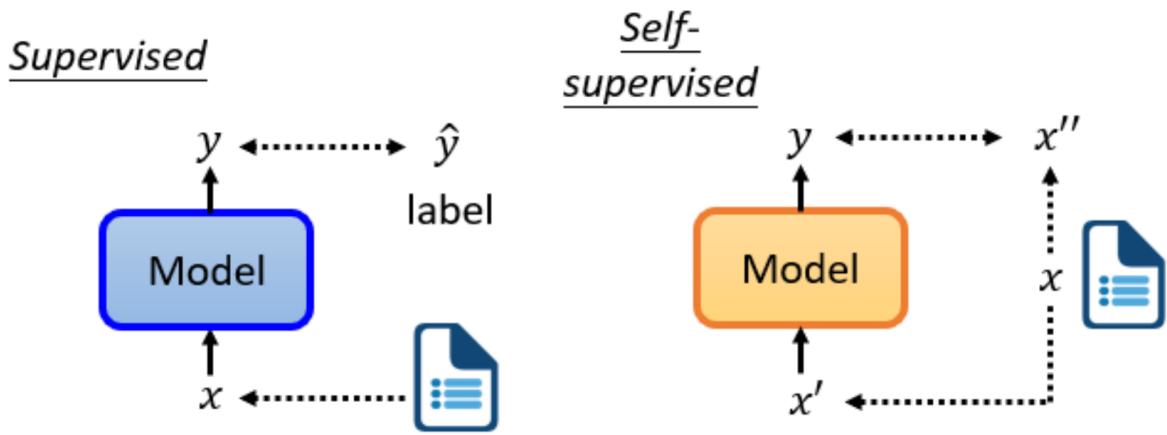
Find unusual data points.

# BERT 简介

---

## Self-supervised Learning

---



 Yann LeCun  
2019年4月30日 · ●

I now call it "self-supervised learning", because "unsupervised" is both a loaded and confusing term.

In self-supervised learning, the system learns to predict part of its input from other parts of its input. In other words a portion of the input is used as a supervisory signal to a predictor fed with the remaining portion of the input.

- 每个人都应该熟悉监督学习，当我们做监督学习时，我们只有一个模型，这个模型的输入是 $x$ ，输出是 $y$ 。

假设你今天想做情感分析，你就是让机器阅读一篇文章，而机器需要对这篇文章进行分类，是正面的还是负面的，你必须先找到大量的文章，你需要对所有的文章进行label。我们需要有标签和文章数据来训练监督模型

- "Self-supervised" 是用另一种方式来监督，没有标签。假设我们只有一堆没有label的文章，但我们试图找到一种方法把它分成两部分。

**我们让其中一部分作为模型的输入数据，另一部分作为标签。**

假设你有没有label的数据，例如，一篇文章叫 $x$ ，我们把 $x$ 分成两部分，一部分叫 $x'$ ，另一部分叫 $x''$ ，我知道现在的说明很抽象。稍后，当我们真正谈论BERT时，你可以更好地理解Self-supervised的含义，以及如何在明明没有办法进行监督训练的情况下，最终还是找到了自己进行监督训练的方法。

我们把 $x$ 分成两部分， $x'$ 和 $x''$ ，然后把 $x'$ 输入模型，让它输出 $y$ 。如果我们在模型训练中使用标签，我们称之为监督学习。由于在Self-supervised学习中不使用标签，我们可以说，Self-supervised学习也是一种无监督的学习方法。但之所以叫Self-supervised Learning，是为了让定义更清晰。

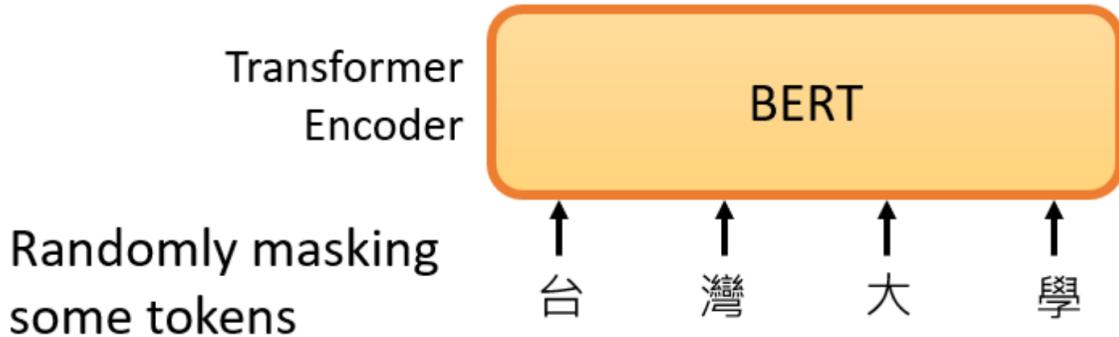
- "Self-supervised Learning"这个词，当初Yann LeCun说过，其实并不是一个老词。根据2019年4月在Facebook上的一个帖子，他说，我现在说的这个方法，他叫Self-supervised Learning。为什么不叫无监督学习呢？因为无监督学习是一个比较大的家族，里面有很多不同的方法，为了让定义更清晰，我们叫它"自监督"，比如我们之前提到的cycle gan，也是无监督学习的一个案例，我们也不使用标注的配对数据，但是，它和Self-supervised Learning还是有点区别。在无监督学习的范畴内，有很多方法，Self-supervised Learning就是其中之一。

# Pre-train

## Masking Input

Self-supervised Learning是什么意思呢，我们直接拿BERT模型来说。

首先，BERT是一个transformer的Encoder，我们已经讲过transformer了，我们也花了很多时间来介绍Encoder和Decoder，transformer中的Encoder实际上是BERT的架构，它和transformer的Encoder完全一样，里面有很多Self-Attention和Residual connection，还有Normalization等等，那么，这就是BERT。

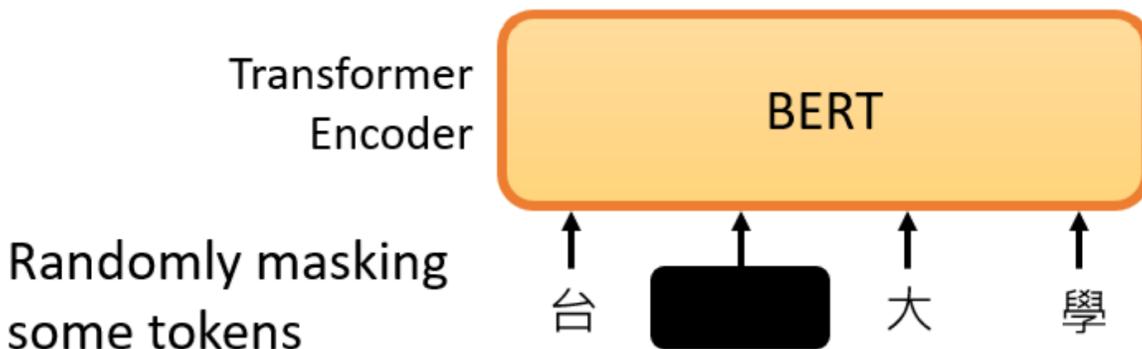


如果你已经忘记了Encoder里有哪些部件，你需要记住的关键点是，BERT可以输入一串向量，然后输出另一串向量，输出的向量个数与输入相同。

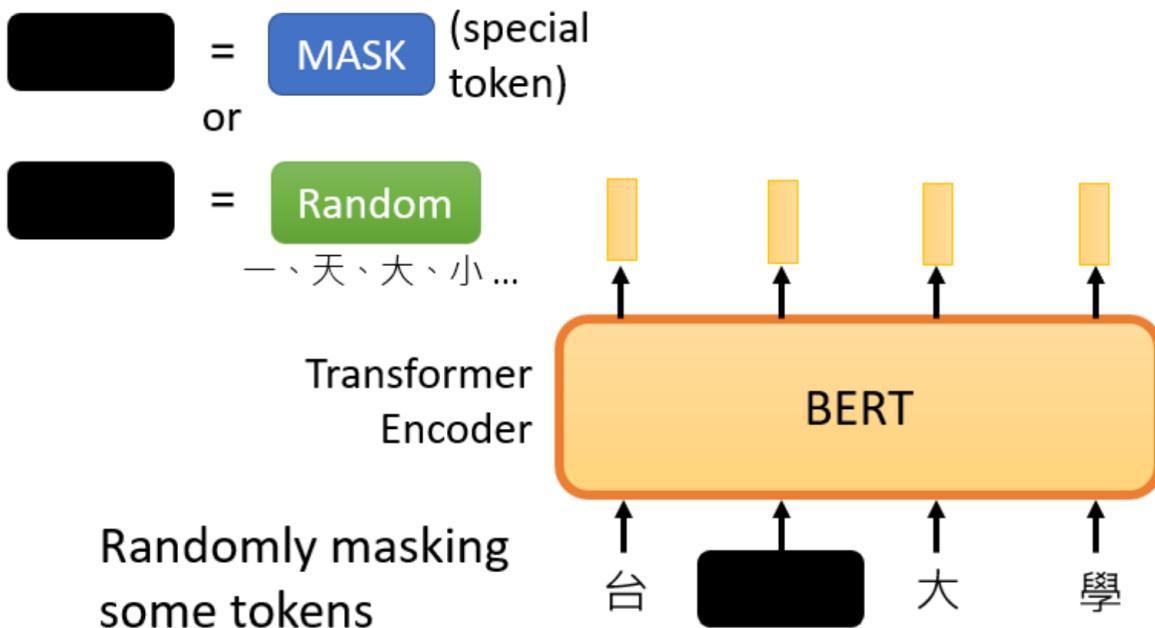
BERT一般用于自然语言处理，用于文本场景，所以一般来说，它的输入是一串文本，也是一串数据。

当我们真正谈论Self-Attention的时候，我们也说不仅文本是一种序列，而且语音也可以看作是一种序列，甚至图像也可以看作是一堆向量。BERT同样的想法是，不仅用于NLP，或者用于文本，它也可以用于语音和视频。

接下来我们需要做的是，随机盖住一些输入的文字，被mask的部分是随机决定的，例如，我们输入100个token，什么是token？在中文文本中，我们通常把一个汉字看作是一个token，当我们输入一个句子时，其中的一些词会被随机mask。



mask的具体实现有两种方法。

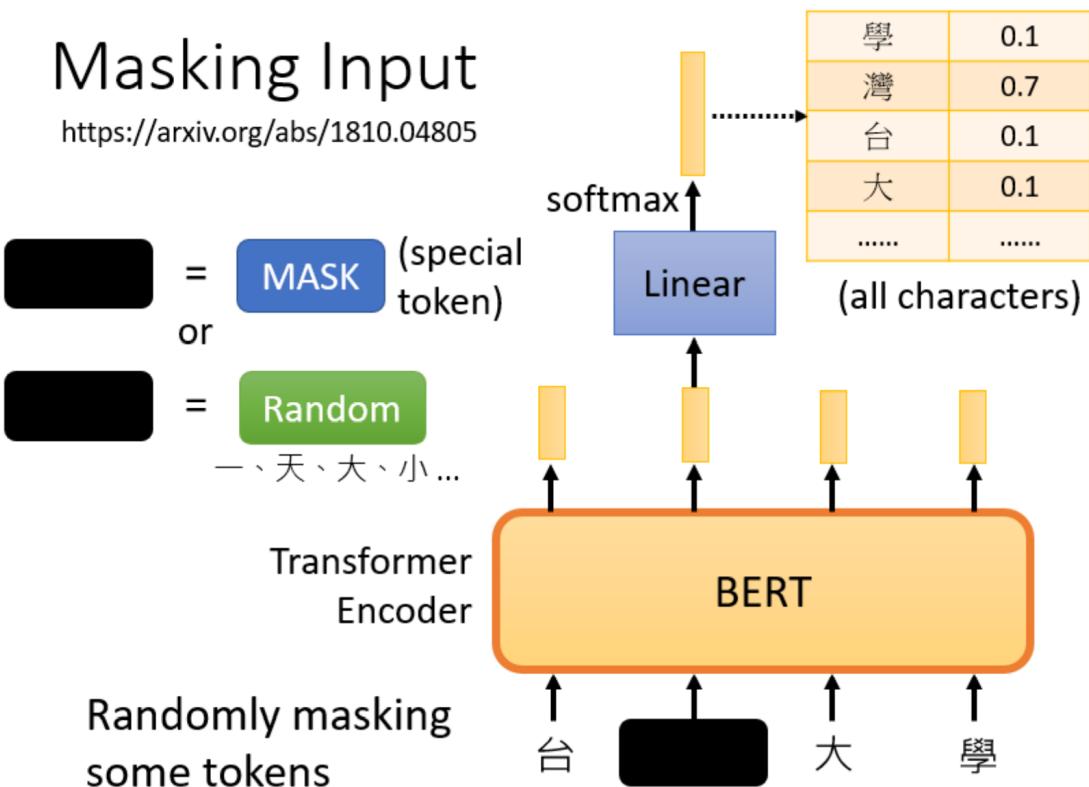


- 第一种方法是，用一个特殊的符号替换句子中的一个词，我们用 "MASK" 标记来表示这个特殊符号，你可以把它看作一个新字，这个字完全是一个新词，它不在你的字典里，这意味着mask了原文。
- 另外一种方法，随机把某一个字换成另一个字。中文的 "湾" 字被放在这里，然后你可以选择另一个中文字来替换它，它可以变成 "一" 字，变成 "天" 字，变成 "大" 字，或者变成 "小" 字，我们只是用随机选择的某个字来替换它

所以有两种方法来做mask，一种是添加一个特殊的标记 "MASK"，另一种是用一个字来替换某个字。

两种方法都可以使用。使用哪种方法也是随机决定的。因此，当BERT进行训练时，向BERT输入一个句子，先随机决定哪一部分的汉字将被mask。

mask后，一样是输入一个序列，我们把BERT的相应输出看作是另一个序列，接下来，我们在输入序列中寻找mask部分的相应输出，然后，这个向量将通过一个Linear transform。



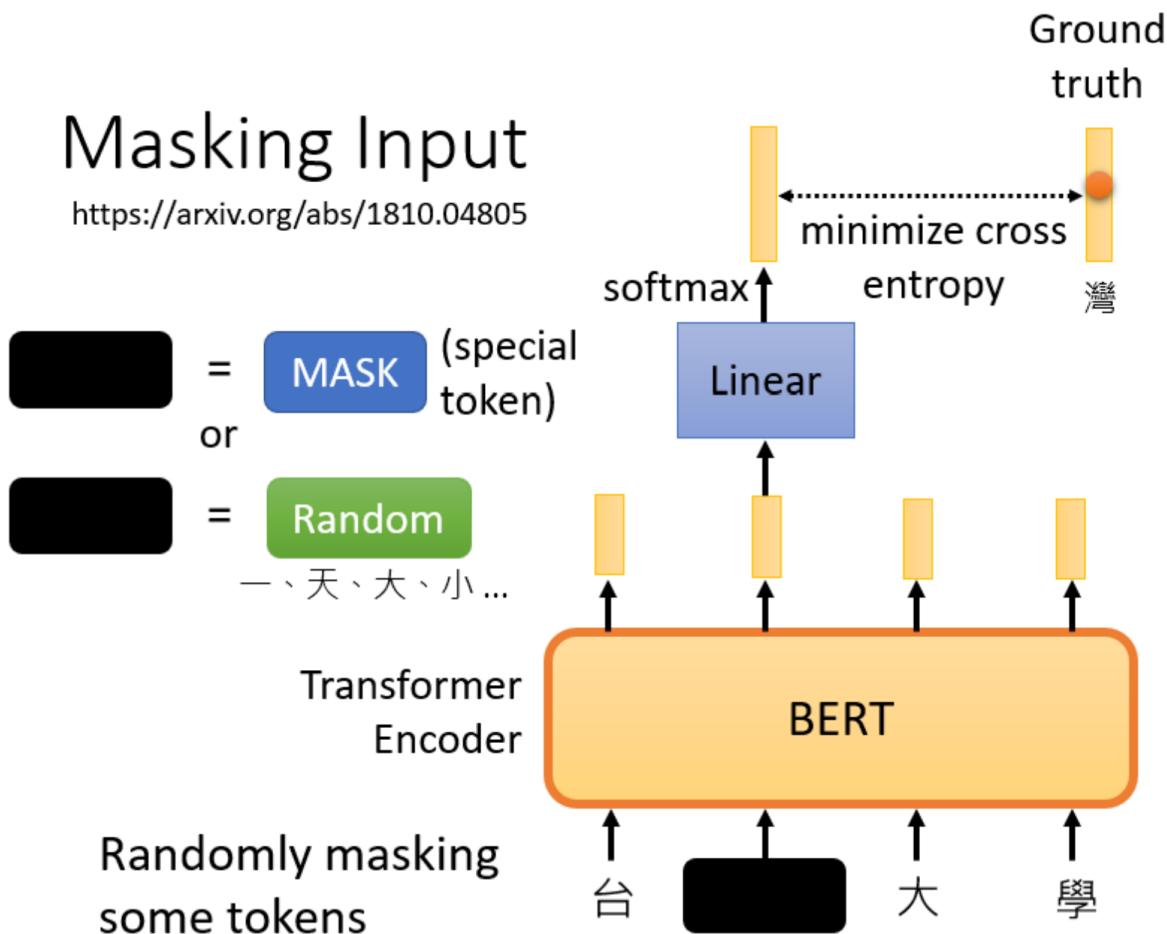
所谓的Linear transform是指，输入向量将与一个矩阵相乘，然后做softmax，输出一个分布。

这与我们在Seq2Seq模型中提到的使用transformer进行翻译时的输出分布相同。输出是一个很长的向量，包含我们想要处理的每个汉字，每一个字都对应到一个分数。

在训练过程中。我们知道被mask的字符是什么，而BERT不知道，我们可以用一个one-hot vector来表示这个字符，并使输出和one-hot vector之间的交叉熵损失最小。

# Masking Input

<https://arxiv.org/abs/1810.04805>



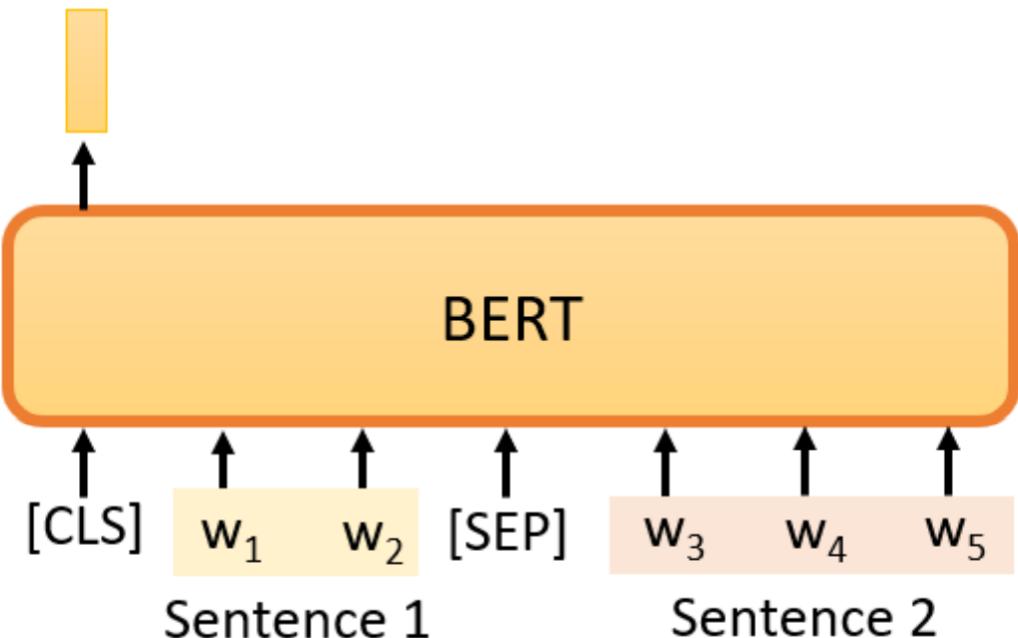
或者说得简单一点，我们实际上是在解决一个分类问题。现在，BERT要做的是，预测什么被盖住。被掩盖的字符，属于 "湾"类。

在训练中，我们在BERT之后添加一个线性模型，并将它们一起训练。所以，BERT里面是一个transformer的Encoder，它有一堆参数。这两个需要共同训练，并试图预测被覆盖的字符是什么，这叫做mask。

## Next Sentence Prediction

事实上，当我们训练BERT时，除了mask之外，我们还会使用另一种方法，这种额外的方法叫做Next Sentence Prediction。

它的意思是，我们从数据库中拿出两个句子，这是我们通过在互联网上抓取和搜索文件得到的大量句子集合，我们在这两个句子之间添加一个特殊标记。这样，BERT就可以知道，这两个句子是不同的句子，因为这两个句子之间有一个分隔符。



我们还将在句子的开头添加一个特殊标记，这里我们用CLS来表示这个特殊标记。

现在，我们有一个很长的序列，包括两个句子，由SEP标记和前面的CLS标记分开。如果我们把它传给BERT，它应该输出一个序列，因为输入也是一个序列，这毕竟是Encoder的目的。

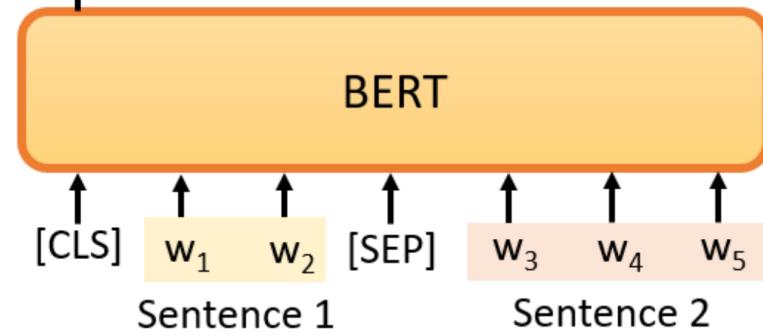
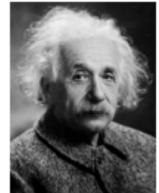
我们将只看CLS的输出，我们将把它乘以一个Linear transform。

- This approach is not helpful.

## Robustly optimized BERT approach (RoBERTa) <https://arxiv.org/abs/1907.11692>

- **SOP:** Sentence order prediction

Used in ALBERT  
<https://arxiv.org/abs/1909.11942>



现在它必须做一个二分类问题，有两个可能的输出：是或不是。这个方法被称为Next Sentence Prediction，所以我们需要预测，第二句是否是第一句的后续句。

然而，后来的研究发现，对于BERT要做的任务来说，Next Sentence Prediction 并没有真正的帮助。例如，有一篇论文叫 "Robustly Optimized BERT Approach"，简称RoBERTa。在这篇论文中，它明确指出，实施Next Sentence Prediction，几乎没有任何帮助。然后，这个概念不知不觉地成为主流。

在这之后，另一篇论文说下一句话预测没有用，所以在它之后的许多论文也开始说它没有用。例如，SCAN-BERT和XLNet都说Next Sentence Prediction 方法是无用的。它可能是无用的原因之一是，Next Sentence Prediction 太简单了，是一项容易的任务。

这个任务的典型方法是，首先随机选择一个句子，然后从数据库中或随机选择要与前一个句子相连的句子。通常，当我们随机选择一个句子时，它看起来与前一个句子有很大不同。对于BERT来说，预测两个句子是否相连并不是太难。因此，在训练BERT完成Next Sentence Prediction 的任务时，没有学到什么太有用的东西。

还有一种类似于Next Sentence Prediction 的方法，它在纸面上看起来更有用，它被称为Sentence order prediction，简称SOP。

这个方法的主要思想是，我们最初挑选的两个句子可能是相连的。可能有两种可能性：要么句子1在句子2后面相连，要么句子2在句子1后面相连。有两种可能性，我们问BERT是哪一种。

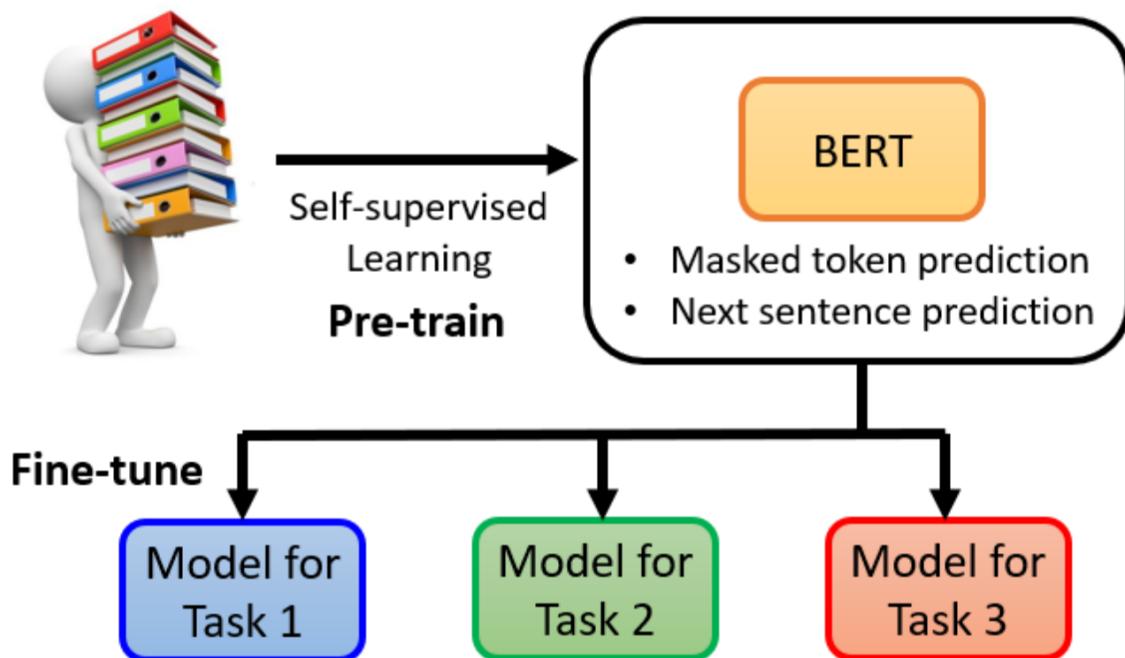
也许因为这个任务更难，它似乎更有效。它被用在一个叫ALBERT的模型中，这是BERT的高级版本。由于ALBERT这个名字与爱因斯坦相似，我在幻灯片中放了一张爱因斯坦的图片。

## Downstream Tasks——Fine-tune

当我们训练时，我们要求BERT学习两个任务。

- 一个是掩盖一些字符，具体来说是汉字，然后要求它填补缺失的字符。
- 另一个任务表明它能够预测两个句子是否有顺序关系。

所以总的来说，BERT它学会了如何填空。BERT的神奇之处在于，在你训练了一个填空的模型之后，它还可以用于其他任务。这些任务不一定与填空有关，也可能是完全不同的任务，但BERT仍然可以用于这些任务，这些任务是BERT实际使用的任务，它们被称为Downstream Tasks(下游任务)，以后我们将谈论一些Downstream Tasks 的例子。



### Downstream Tasks

- The tasks we care
- We have a little bit labeled data.

所谓的 "Downstream Tasks "是指，你真正关心的任务。但是，当我们想让BERT学习做这些任务时，我们仍然需要一些标记的信息。

总之，BERT只是学习填空，但是，以后可以用来做各种你感兴趣的Downstream Tasks 。它就像胚胎中的干细胞，它有各种无限的潜力，虽然它还没有使用它的力量，它只能填空，但以后它有能力解决各种任务。我们只需要给它一点数据来激发它，然后它就能做到。

BERT分化成各种任务的功能细胞，被称为Fine-tune(微调)。所以，我们经常听到有人说，他对BERT进行了微调，也就是说他手上有一个BERT，他对这个BERT进行了微调，使它能够完成某种任务，与微调相反，在微调之前产生这个BERT的过程称为预训练。

所以，生成BERT的过程就是Self-supervised学习。但是，你也可以称之为预训练。如果你知道它是什么，你不应该在其他地方寻找Self-supervised学习的模型，直接应用在作业上。因为这些方法，往往带有令人难以置信的强大能力，这将使你要做的事情变得很无聊。

接下来其实还有一个BERT的作业。作业7是使用BERT。所以，在作业7中，当然，你可以使用预训练的模型。这是你唯一可以使用预训练模型的作业，因为作业7是微调BERT。所以，你当然要使用预训练的BERT，来进行微调。所以，只有在作业7中，你可以使用预训练的模型。

好的，在我们谈论如何微调BERT之前，我们应该先看看它的能力。今天，为了测试Self-supervised学习的能力，通常，你会在多个任务上测试它。因为我们刚才说，BERT就像一个胚胎干细胞，它要分化成各种任务的功能细胞，我们通常不会只在一个任务上测试它的能力，你会让这个BERT分化成各种任务的功能细胞，看看它在每个任务上的准确性，然后我们取其平均值，得到一个总分。这种不同任务的集合，我们可以称之为任务集。**任务集中最著名的基准被称为GLUE，它是General Language Understanding Evaluation的缩写。**

## GLUE

General Language Understanding  
Evaluation (GLUE)

<https://gluebenchmark.com/>

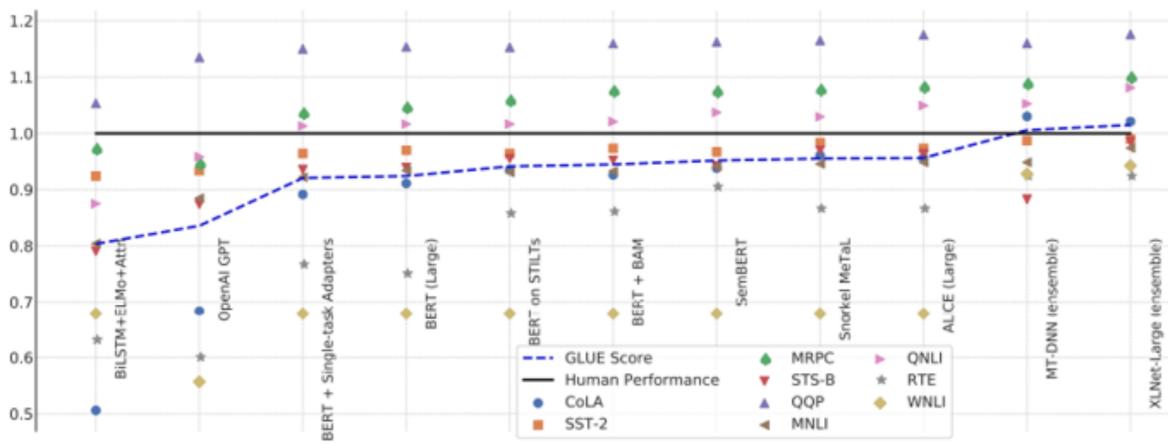
- Corpus of Linguistic Acceptability (CoLA)
- Stanford Sentiment Treebank (SST-2)
- Microsoft Research Paraphrase Corpus (MRPC)
- Quora Question Pairs (QQP)
- Semantic Textual Similarity Benchmark (STS-B)
- Multi-Genre Natural Language Inference (MNLI)
- Question-answering NLI (QNLI)
- Recognizing Textual Entailment (RTE)
- Winograd NLI (WNLI)

GLUE also has Chinese version (<https://www.cluebenchmarks.com/>)

在GLUE中，总共有9个任务。一般来说，你想知道像BERT这样的模型是否被训练得很好。所以，你实际上会得到9个模型，用于9个单独的任务。你看看这9个任务的平均准确率，然后，你得到一个值。这个值代表这个Self-supervised模型的性能。

让我们看看BERT在GLUE上的性能。

## • GLUE scores



Source of image: <https://arxiv.org/abs/1905.00537>

有了BERT，GLUE得分，也就是9个任务的平均得分，确实逐年增加。在这张图中，,横轴表示不同的模型，这里列出了，你可以发现，除了ELMO和GPT，其他的还有很多BERT，各种BERT。

黑色的线，表示人类的工作，也就是人类在这个任务上的准确度，那么，我们把这个当作1，这里每一个点代表一个任务，那么，你为什么要和人类的准确度进行比较呢？

人类的准确度是1，如果他们比人类好，这些点的值就会大于1，如果他们比人类差，这些点的值就会小于1，这是因为这些任务，其评价指标可能不是准确度。每个任务使用的评价指标是不同的，它可能不是准确度。如果我们只是比较它们的值，可能是没有意义的。所以，这里我们看的是人类之间的差异。

所以，你会发现，在原来的9个任务中，只有1个任务，机器可以比人类做得更好。随着越来越多的技术被提出，越来越多的,还有3个任务可以比人类做得更好。对于那些远不如人类的任务，,它们也在逐渐追赶。

蓝色曲线表示机器GLUE得分的平均值。还发现最近的一些强势模型，例如XLNET，甚至超过了人类。当然，这只是这些数据集的结果，并不意味着机器真的在总体上超过了人类。它在这些数据集上超过了人类。这意味着这些数据集并不能代表实际的表现，而且难度也不够大。

所以，在GLUE之后，有人做了Super GLUE。他们找到了更难的自然语言处理任务，让机器来解决。好了！展示这幅图的意义主要是告诉大家，有了BERT这样的技术，机器在自然语言处理方面的能力确实又向前迈进了一步。

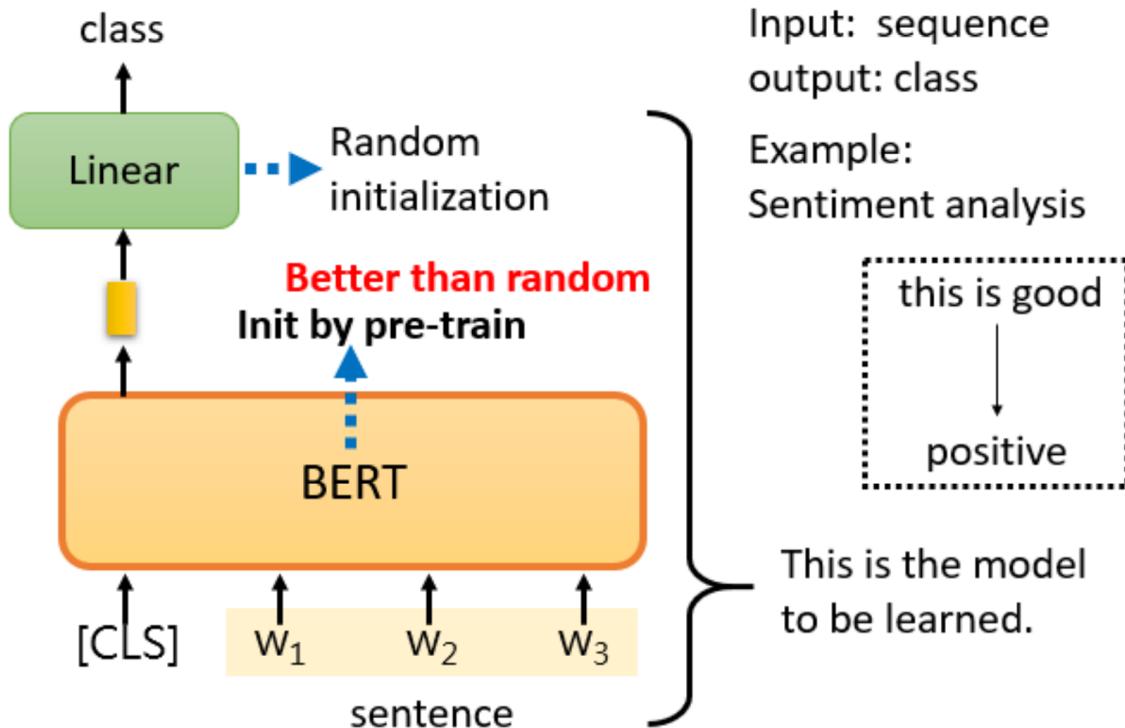
BERT到底是怎么用的呢？我们将给出4个关于BERT的应用案例，

## How to use BERT

### Case 1: Sentiment analysis

第一个案例是这样的，我们假设我们的Downstream Tasks 是输入一个序列，然后输出一个class，这是一个分类问题。

比如说Sentiment analysis情感分析，就是给机器一个句子，让它判断这个句子是正面的还是负面的。



对于BERT来说，它是如何解决情感分析的问题的？

你只要给它一个句子，也就是你想用它来判断情绪的句子，然后把CLS标记放在这个句子的前面，我刚才提到了CLS标记。我们把CLS标记放在前面，扔到BERT中，这4个输入实际上对应着4个输出。然后，我们只看CLS的部分。CLS在这里输出一个向量，我们对它进行Linear transform，也就是将它乘以一个Linear transform的矩阵，这里省略了Softmax。

然而，在实践中，你必须为你的Downstream Tasks 提供标记数据，换句话说，BERT没有办法从头开始解决情感分析问题，你仍然需要向BERT提供一些标记数据，你需要向它提供大量的句子，以及它们的正负标签，来训练这个BERT模型。

在训练的时候，Linear transform和BERT模型都是利用Gradient descent来更新参数的。

- Linear transform的参数是随机初始化的
- 而BERT的参数是由学会填空的BERT初始化的。

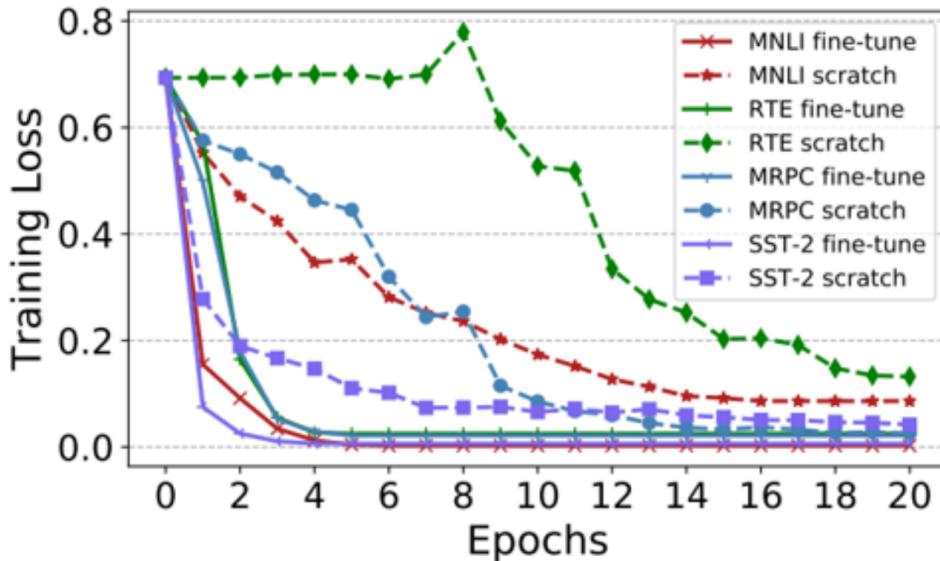
每次我们训练模型的时候，我们都要初始化参数，我们利用梯度下降来更新这些参数，然后尝试 minimize loss，

例如，我们正在做情感分类，但是，我们现在有BERT。我们不必随机初始化所有的参数。,我们唯一随机初始化的部分是Linear这里。BERT的骨干是一个巨大的transformer的Encoder。这个网络的参数不是随机初始化的。把学过填空的BERT参数，放到这个地方的BERT中作为参数初始化。

为什么要这样做呢？为什么要用学过填空的BERT，再放到这里呢？最直观和最简单的原因是，它比随机初始化新参数的网络表现更好。当你把学会填空的BERT放在这里时，它将获得比随机初始化BERT更好的性能。

# Pre-train v.s. Random Initialization

(fine-tune) (scratch)



Source of image: <https://arxiv.org/abs/1908.05620>

在这里有篇文章中有一个例子。横轴是训练周期，纵轴是训练损失，到目前为止，大家对这种图一定很熟悉，随着训练的进行，损失当然会越来越低，这个图最有趣的地方是，有各种任务。我们不会解释这些任务的细节，我只想说明有各种任务。

- "fine-tune"是指模型被用于预训练，这是网络的BERT部分。该部分的参数是由学习到的BERT的参数来初始化的，以填补空白。
- scratch表示整个模型，包括BERT和Encoder部分都是随机初始化的。

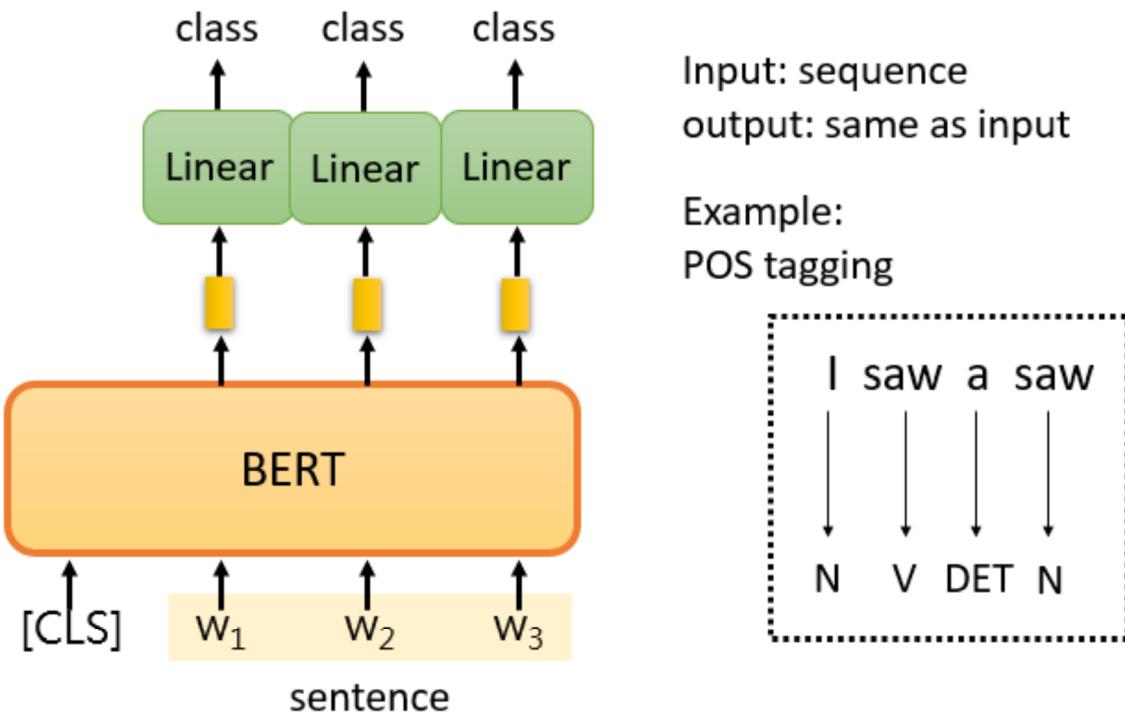
首先，在训练网络时，scratch与用学习填空的BERT初始化的网络相比，损失下降得比较慢，最后，用随机初始化参数的网络的损失仍然高于用学习填空的BERT初始化的参数。

- 当你进行Self-supervised学习时，你使用了大量的无标记数据。
- 另外，Downstream Tasks 需要少量的标记数据。

所谓的 "半监督 "是指，你有大量的无标签数据和少量的有标签数据，这种情况被称为 "半监督"，所以使用BERT的整个过程是连续应用Pre-Train和Fine-Tune，它可以被视为一种半监督方法。

## Case 2 : POS tagging

第二个案例是，输入一个序列，然后输出另一个序列，而输入和输出的长度是一样的。我们在讲Self-Attention的时候，也举了类似的例子。例如，POS tagging。



POS tagging的意思是词性标记。你给机器一个句子，它必须告诉你这个句子中每个词的词性，即使这个词是相同的，也可能有不同的词性。

你只需向BERT输入一个句子。之后，对于这个句子中的每一个标记，它是一个中文单词，有一个代表这个单词的相应向量。然后，这些向量会依次通过Linear transform和Softmax层。最后，网络会预测给定单词所属的类别，例如，它的词性。

当然，类别取决于你的任务，如果你的任务不同，相应的类别也会不同。接下来你要做的事情和案例1完全一样。换句话说，你需要有一些标记的数据。这仍然是一个典型的分类问题。唯一不同的是，BERT部分，即网络的Encoder部分，其参数不是随机初始化的。在预训练过程中，它已经找到了不错的参数。

当然，我们在这里展示的例子属于自然语言处理。但是，你可以把这些例子改成其他任务，例如，你可以把它们改成语音任务，或者改成计算机视觉任务。我在Self-supervised Learning一节中提到，语音、文本和图像都可以表示为一排向量。虽然下面的例子是文字，但这项技术不仅限于处理文字，它还可以用于其他任务，如计算机视觉。

### Case 3: Natural Language Inference

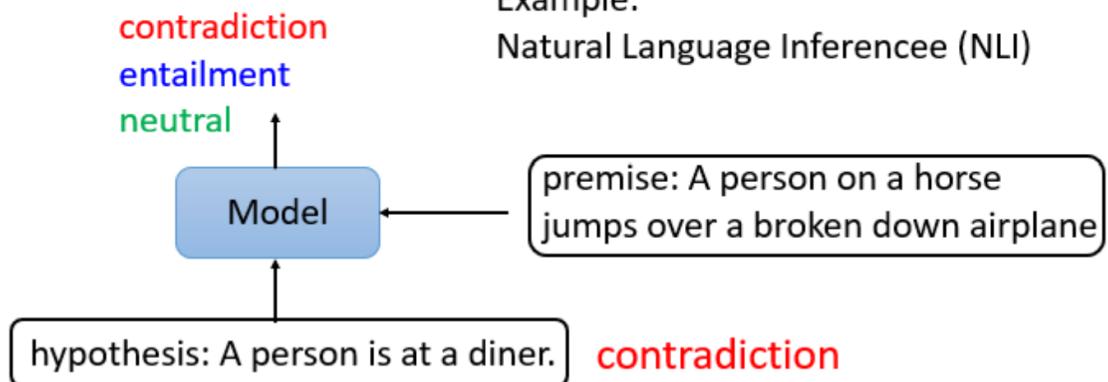
在案例3中，模型输入两个句子，输出一个类别。好了，第三个案例以两个句子为输入，输出一个类别，什么样的任务采取这样的输入和输出？最常见的是Natural Language Inference，它的缩写是NLI

Input: two sequences

Output: a class

Example:

Natural Language Inferencee (NLI)



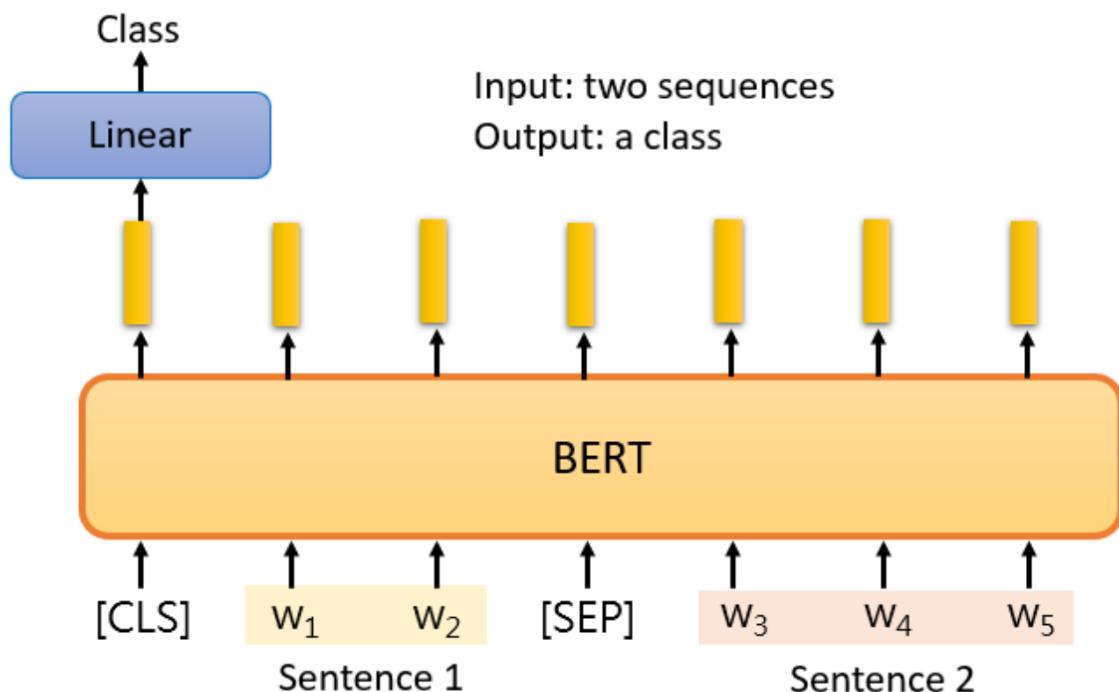
机器要做的是判断，是否有可能从前提中推断出假设。这个前提与这个假设相矛盾吗？或者说它们不是相矛盾的句子？

在这个例子中，我们的前提是，一个人骑着马，然后他跳过一架破飞机，这听起来很奇怪。但这个句子实际上是这样的。这是一个基准语料库中的例子。

这里的假设是，这个人在一个餐馆。所以推论说这是一个矛盾。

所以机器要做的是，把两个句子作为输入，并输出这两个句子之间的关系。这种任务很常见。它可以用在哪里呢？例如，舆情分析。给定一篇文章，下面有一个评论，这个消息是同意这篇文章，还是反对这篇文章？该模型想要预测的是每条评论的位置。事实上，有很多应用程序接收两个句子，并输出一个类别。

BERT是如何解决这个问题的？你只要给它两个句子，我们在这两个句子之间放一个特殊的标记，并在最开始放CLS标记。



这个序列是BERT的输入。但我们只把CLS标记作为Linear transform的输入。它决定这两个输入句子的类别。对于NLI，你必须问，这两个句子是否是矛盾的。它是用一些预先训练好的权重来初始化的。

## Case 4: Extraction-based Question Answering (QA)

如果你不理解前面的案例，就忘掉它们。这第四个案例，就是我们在作业7中要做的。作业7是一个问题回答系统。也就是说，在机器读完一篇文章后，你问它一个问题，它将给你一个答案。

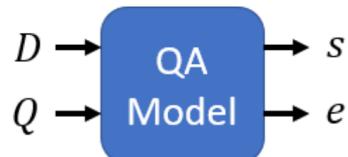
但是，这里的问题和答案稍有限制。这是Extraction-based的QA。也就是说，我们假设答案必须出现在文章中。答案必须是文章中的一个片段。

在这个任务中，一个输入序列包含一篇文章和一个问题，文章和问题都是一个序列。对于中文来说，每个d代表一个汉字，每个q代表一个汉字。你把d和q放入QA模型中，我们希望它输出两个正整数s和e。根据这两个正整数，我们可以直接从文章中截取一段，它就是答案。这个片段就是正确的答案。

- Extraction-based Question Answering (QA)

**Document:**  $D = \{d_1, d_2, \dots, d_N\}$

**Query:**  $Q = \{q_1, q_2, \dots, q_M\}$



output: two integers ( $s, e$ )

**Answer:**  $A = \{d_s, \dots, d_e\}$

In meteorology, precipitation is any product of the condensation of atmospheric water vapor that falls under **gravity**. The main forms of precipitation include drizzle, rain, sleet, snow, **graupel** and hail... Precipitation forms as smaller droplets coalesce via collision with other rain drops or ice crystals **within a cloud**. Short, intense periods of rain in scattered locations are called "showers".

What causes precipitation to fall?  
**gravity**

What is another main form of precipitation besides drizzle, rain, snow, sleet and hail?  
**graupel**

Where do water droplets collide with ice crystals to form precipitation?  
**within a cloud**

这听起来很疯狂，但是，这是现在使用的一个相当标准的方法。六年前，当我第一次听说这个机制可以解决QA任务时，我简直不敢相信。但是，无论如何，这是今天一个非常普遍的方法。

好吧，如果你仍然不明白我在说什么，更具体地说，这里有一个问题和一篇文章，正确答案是 "gravity"。机器如何输出正确答案？

In meteorology, precipitation is any product of the condensation of atmospheric water vapor that falls under **gravity**. The main forms of precipitation include drizzle, rain, sleet, snow, **graupel** and hail... Precipitation forms as smaller droplets coalesce via collision with other rain drops or ice crystals **within a cloud**. Short, intense periods of rain **77** atte **79** cations are called "showers".

What causes precipitation to fall?

**gravity**

$$s = 17, e = 17$$

What is another main form of precipitation besides drizzle, rain, snow, sleet and hail?

**graupel**

Where do water droplets collide with ice crystals to form precipitation?

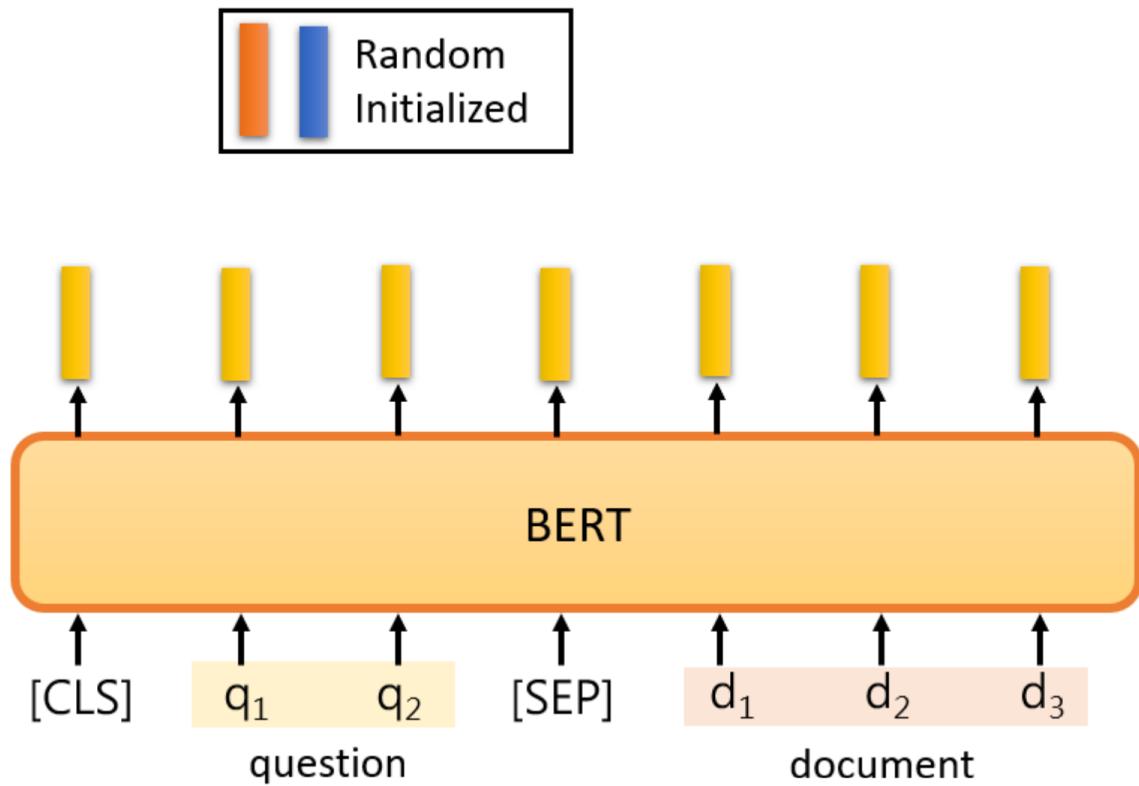
**within a cloud**

$$s = 77, e = 79$$

你的保证模型应该输出， $s$ 等于17， $e$ 等于17，来表示**gravity**。因为它是整篇文章中的第17个词，所以 $s$ 等于17， $e$ 等于17，意味着输出第17个词作为答案。

或者举另一个例子，答案是，"within a cloud"，这是文章中的第77至79个词。你的模型要做的是，输出77和79这两个正整数，那么文章中从第77个词到第79个词的分割应该是最终的答案。这就是作业7要你做的。

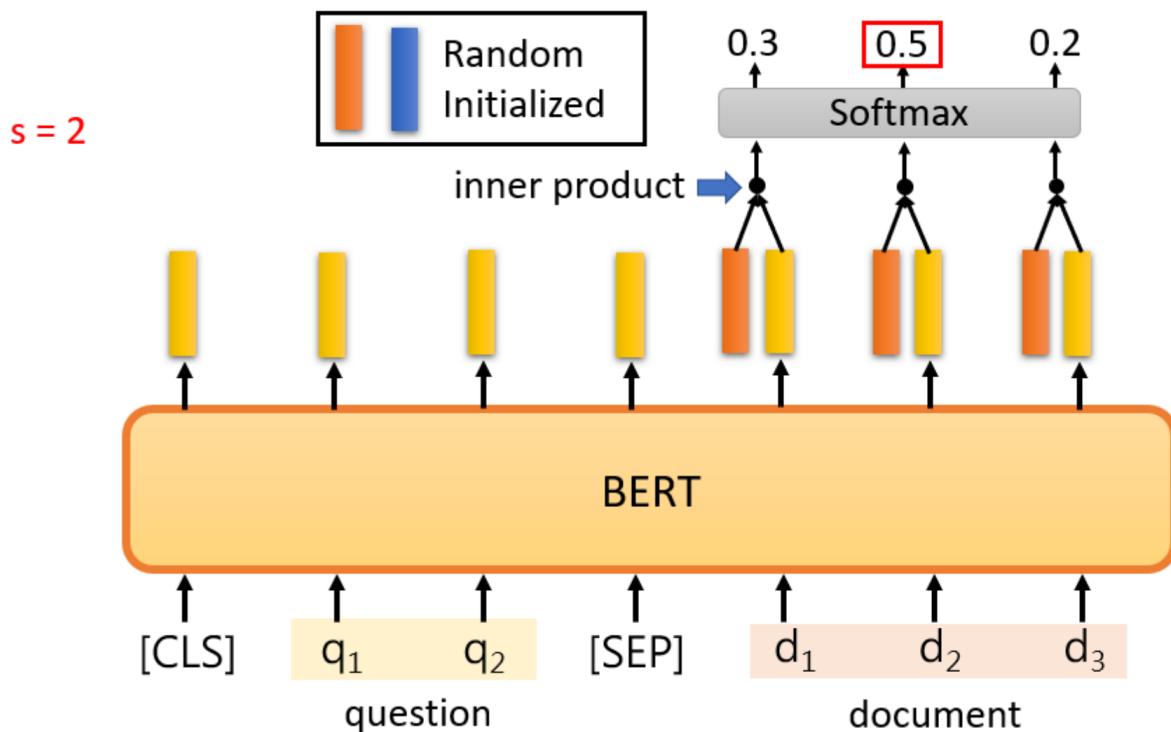
当然，我们不是从头开始训练QA模型，为了训练这个QA模型，我们使用BERT预训练的模型。



这个解决方案是这样的。对于BERT来说，你必须向它展示一个问题，一篇文章，以及在问题和文章之间的一个特殊标记，然后我们在开头放一个CLS标记。

**在这个任务中，你唯一需要从头训练的只有两个向量。“从头训练”是指随机初始化。**这里我们用橙色向量和蓝色向量来表示，这两个向量的长度与BERT的输出相同。

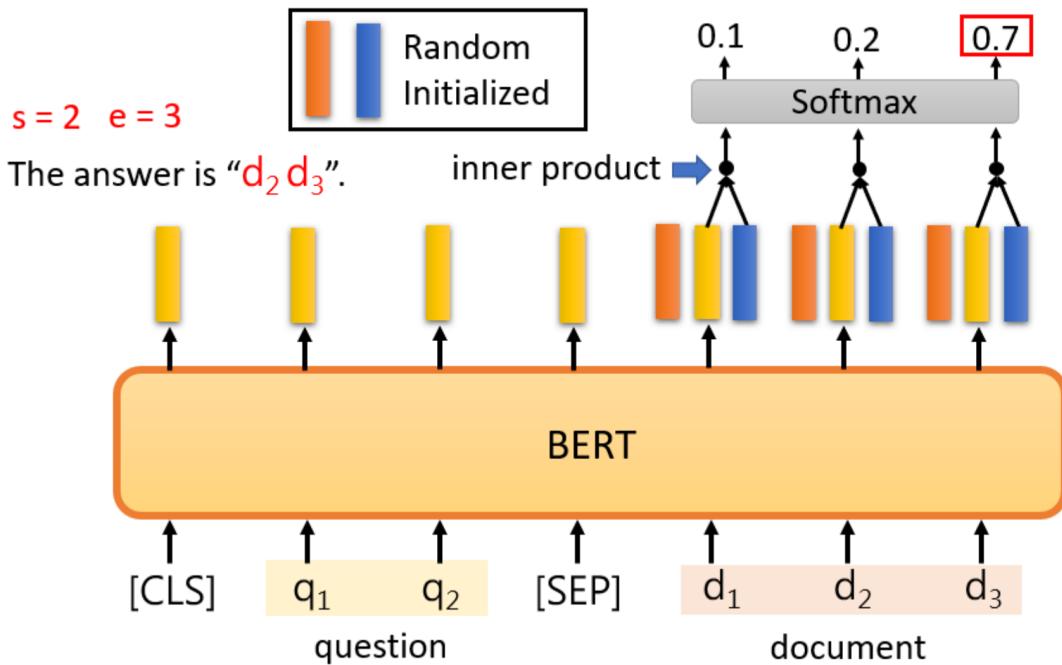
假设BERT的输出是768维的向量，这两个向量也是768维的向量。那么，如何使用这两个向量？



- 首先,计算这个橙色向量和那些与文件相对应的输出向量的内积,由于有3个代表文章的标记,它将输出三个向量,计算这三个向量与橙色向量的内积,你将得到三个值,然后将它们通过softmax函数,你将得到另外三个值。

这个内积和attention很相似，你可以把橙色部分看成是query，黄色部分看成是key，这是一个attention，那么我们应该尝试找到分数最大的位置，就是这里，橙色向量和d<sub>2</sub>的内积，如果这是最大值，s应该等于2，你输出的起始位置应该是2

- 蓝色部分做的是完全一样的事情。



蓝色部分代表答案的终点，我们计算这个蓝色向量与文章对应的黄色向量的内积，然后，我们在这里也使用softmax，最后，找到最大值，如果第三个值是最大的，e应该是3，正确答案是d2和d3。

因为答案必须在文章中，如果答案不在文章中，你就不能使用这个技巧。这就是一个QA模型需要做的。注意，这两个向量是随机初始化的，而BERT是通过它预先训练的权重初始化的。

## Q&A

**Q:** BERT的输入长度有限制吗？

**A:** 理论上，没有。在现实中，是的，在理论上，因为BERT模型，是一个transformer的Encoder，所以它可以输入很长的序列，只要你能够做Self-Attention，但Self-Attention的计算复杂性是非常高的。所以你会发现，在实践中，BERT实际上不能输入太长的序列，你最多可以输入512长度的序列，如果你输入一个512长度的序列，Self-Attention在中间就要产生512乘以512大小的Attention Metric，那么你可能会被计算所淹没。所以实际上它的长度不是无限的。在助教的程序中，已经为大家处理了这个问题。我们限制了BERT的输入长度，而且用一篇文章来训练需要很长的时间。然后每次，我们只取其中的一段进行训练。我们不会将整篇文章输入BERT。因为你想要的距离太长了，你的训练会有问题。

**Q:** "它与填空题有什么关系？

**A:** ",哇，这个问题很好。,你会认为这个填空题只是一个填空题。但我要在这里做一个Q&A。,这两件事之间有什么关系呢？这里先卖个关子，待会会试着回答你。

## Training BERT is challenging!

BERT是这样一个著名的模型，它可以做任何事情，那么你可能会认为BERT，在预训练中，它只是填空题，但是，你自己真的不能把它训练起来。

首先，谷歌最早的BERT，它使用的数据规模已经很大了，它的数据中包含了30亿个词汇，30亿个词汇有多少？是《哈利波特全集》的3000倍。,《哈利波特全集》大约是100万个词汇。,那么谷歌在训练BERT时，最早的BERT，它使用的数据量是《哈利波特全集》的3000倍。

所以你处理起来会比较痛苦,更痛苦的是训练过程,为什么我知道训练过程是痛苦的呢,因为我们实验室有一个学生,他其实是助教之一,他自己试着训练一个BERT,他觉得他不能重现谷歌的结果,好,那么在这个图中,纵轴代表GLUE分数,我们刚才讲到GLUE,对吧?有9个任务,平均有9个任务,平均分数就叫GLUE分数,好的,那么蓝线就是,谷歌原来的BERT的GLUE分数。

那么我们的目标其实不是实现BERT,我们的目标是实现ALBERT。ALBERT是一个高级版本,是橙色的线,蓝线是我们自己训练的ALBERT,但是我们实际训练的不是最大版本,BERT有一个base版本和一个large版本。对于大版本,我们很难自己训练它,所以我们尝试用最小的版本来训练,看它是否与谷歌的结果相同。

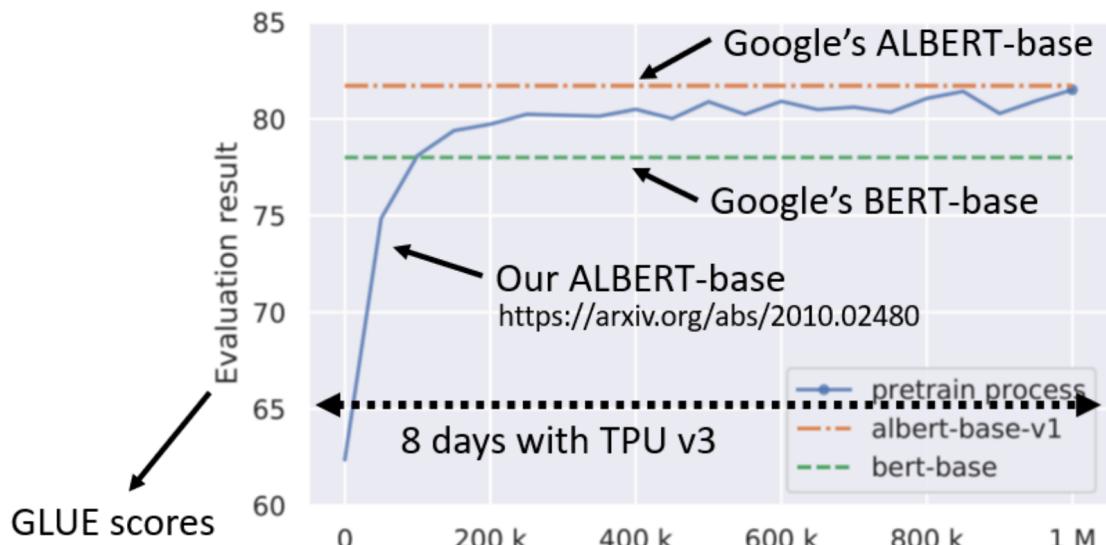
你可能会说这30亿个数据,30亿个词似乎有很多数据。实际上,因为它是无标签数据,所以你只是从互联网上整理了一堆文本,有相同的信息量。所以你要爬上这个级别的信息并不难,难的是训练过程

台達電產學合作計畫研究成果  
This work is done by 姜成翰

## Training BERT is challenging!

Training data has more than **3 billions** of words.

**3000 times of Harry Potter series**



好的,这个横轴是训练过程,参数更新多少次,大约一百万次的更新,需要多长时间,用TPU运行8天,所以你的TPU要运行8天,如果你在Colab上做,这个至少要运行200天,你甚至可能到明年才能得到结果。

所以,你真的很难自己训练这种BERT模型。幸运的是,作业只是对它进行微调。你可以在Colab上进行微调,在Colab上微调BERT只需要半小时到一小时。但是,如果你想从头开始训练它,也就是说,训练它做填空题,这将需要大量的时间,而且,你不能在Colab上自己完成它。

## BERT Embryology (胚胎學)

谷歌已经训练了BERT,而且这些Pre-Train模型是公开的,我们自己训练一个,结果和谷歌的BERT差不多,这有什么意义呢?

其实是想建立BERT胚胎学。“BERT胚胎学是什么意思?”

# BERT Embryology (胚胎學)

<https://arxiv.org/abs/2010.02480>



When does BERT know POS tagging,  
syntactic parsing, semantics?

The answer is counterintuitive!

我们知道在BERT的训练过程中需要非常大的计算资源，所以我们想知道有没有可能，节省这些计算资源？有没有可能让它训练得更快？要知道如何让它训练得更快，也许我们可以从观察它的训练过程开始。

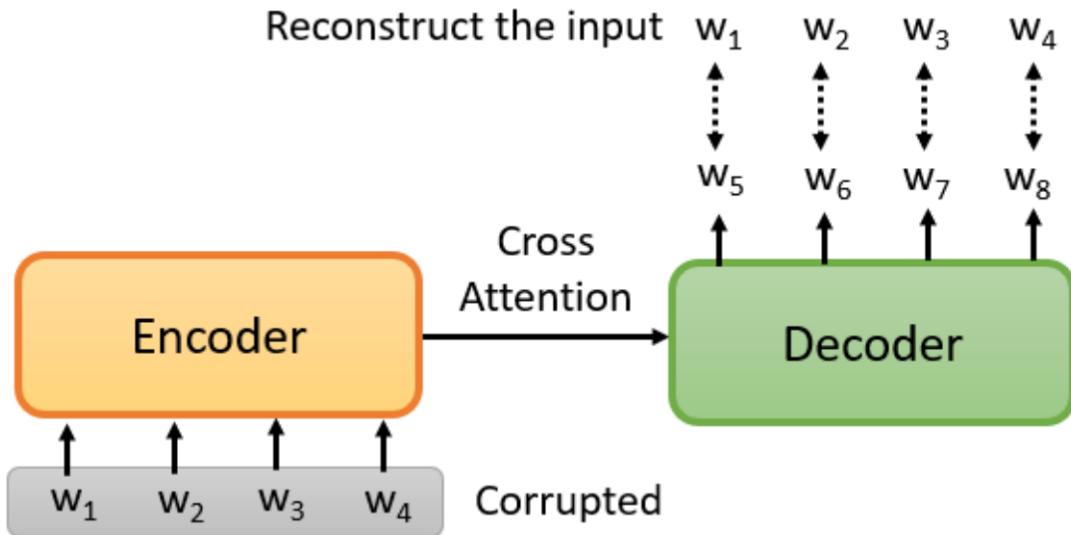
过去没有人观察过BERT的训练过程。因为在谷歌的论文中，他们只是告诉你，我有这个BERT。然后它在各种任务中做得很好。

BERT在学习填空的过程中，学到了什么？"它在这个过程中何时学会填动词？什么时候学会填名词？什么时候学会填代词？没有人研究过这个问题。

所以我们要自己训练BERT后，可以观察到BERT什么时候学会填什么词汇，它是如何提高填空能力的？好了，细节不是这门课的重点，所以我不在这里讲了。我把论文的链接<https://arxiv.org/abs/2010.02480>放在这里，供大家参考。不过可以提前爆冷一下就是：事实和你直观想象的不一样。

## Pre-training a seq2seq model

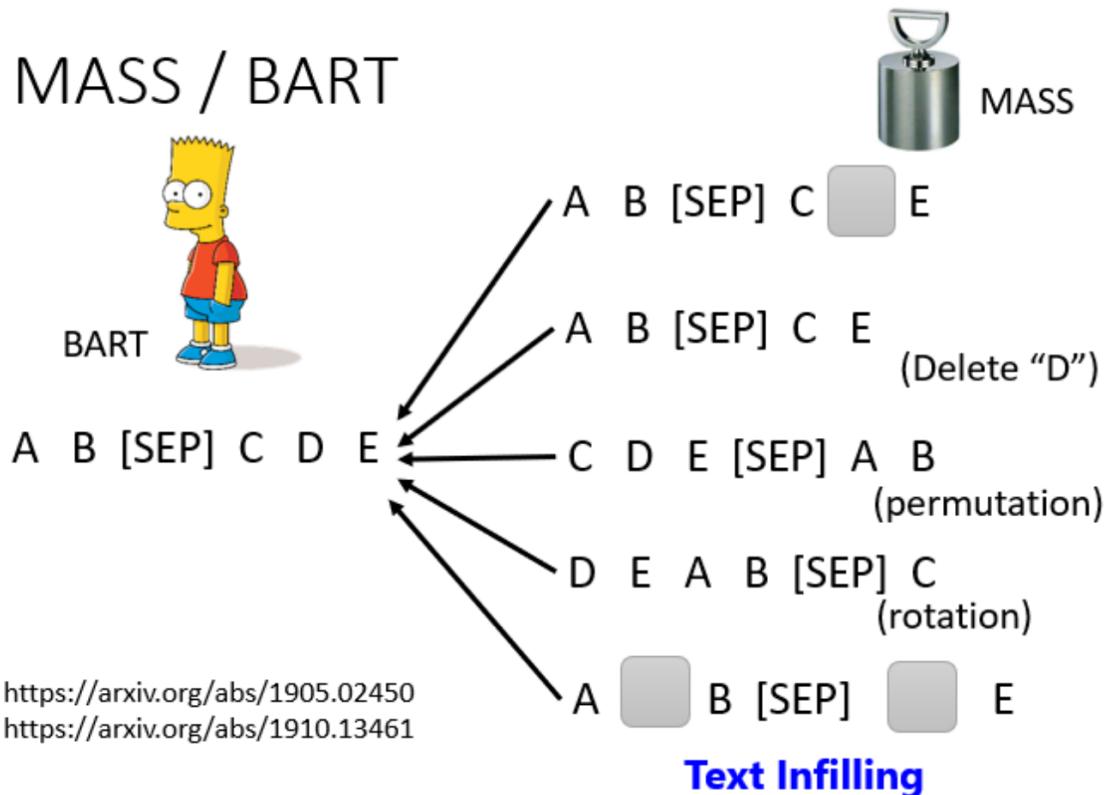
我们补充一点，上述的任务都不包括，Seq2Seq模型，如果我们要解决，Seq2Seq模型呢？BERT只是一个预训练Encoder，有没有办法预训练Seq2Seq模型的Decoder？



有，你就说我有一个Seq2Seq模型，有一个transformer，还有一个Encoder和Decoder。输入是一串句子，输出是一串句子，中间用Cross Attention连接起来，然后你故意在Encoder的输入上做一些**干扰来破坏它**，我以后会具体告诉你我说的 "破坏" 是什么意思

Encoder看到的是被破坏的结果，那么Decoder应该输出句子被破坏前的结果，训练这个模型实际上是预训练一个Seq2Seq模型。

有一篇论文叫MASS



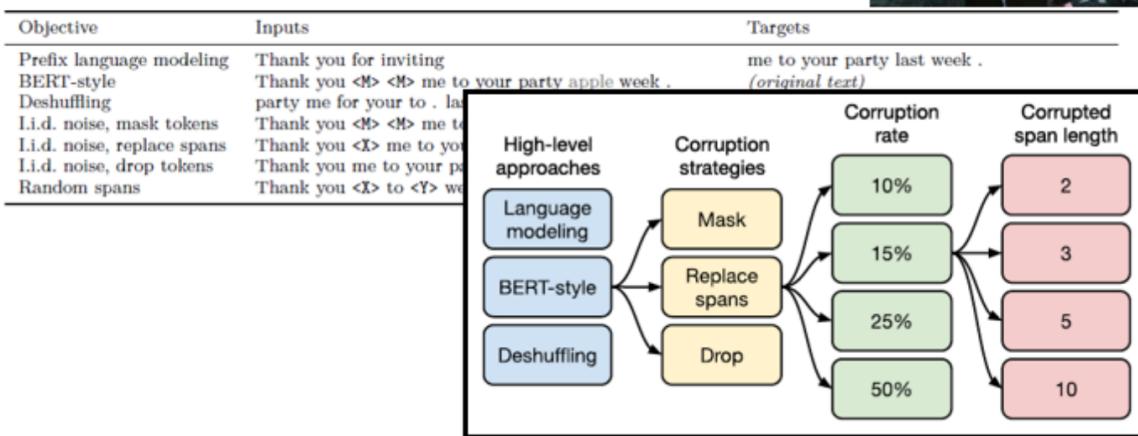
在MASS中，它说破坏的方法是，就像BERT做的那样，只要遮住一些地方就可以了，然后有各种方法来破坏它，比如，删除一些词，打乱词的顺序，旋转词的顺序。或者插入一个MASK，再去掉一些词。总之，有各种方法。在破坏了输入的句子之后，它可以通过Seq2Seq模型来恢复它。

有一篇论文叫BART，它就是用了所有这些方法。我发现用所有可能的方法更好，它可以比MASS更好。我想问一个问题，为什么不是芝麻街的人物？

你可能会问,有那么多的mask方法,哪种方法更好呢?也许你想自己做一些实验来试试,让我告诉你,你不需要做,谷歌为你做的,有一篇论文叫T5。

## T5 – Comparison

- Transfer Text-to-Text Transformer (T5)
- Colossal Clean Crawled Corpus (C4)



T5的全称是Transfer Text-To-Text Transformer, 有五个T, 所以叫T5。在这个T5里面, 它只是做了各种尝试, 它做了你能想象的所有组合。这篇论文有67页, 你可以回去读一下, 看看结论。

T5是在一个语料库上训练的, 叫 "Colossal Clean Crawled Corpus", 对于这个数据集, Colossal就是巨无霸, 就是非常巨大的意思, 它叫C4, 你用C4来训练T5, 大家都是命名高手。这个命名非常强大, 这个C4有多大?

C4是一个公共数据集, 你可以下载它, 它是公共的, 但是它的原始文件大小是7TB, 你可以下载它, 但是你不知道把它保存在哪里, 加载之后, 你可以通过脚本做预处理, 由谷歌提供。这个脚本有一个文件, 我看到它在网站上发布了, 语料库网站上的文件说, 用一个GPU做预处理需要355天, 你可以下载它, 但你在预处理时有问题。

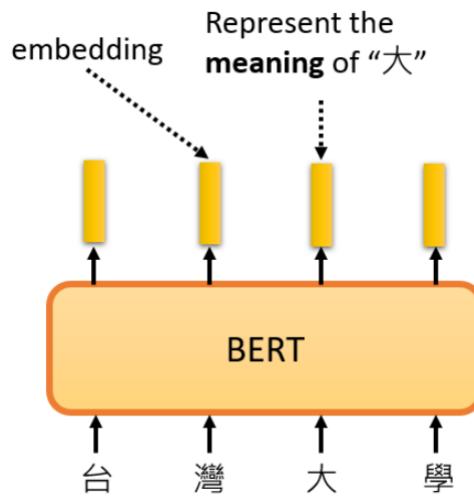
所以, 你可以发现, 在深度学习中, 数据量和模型都很惊人。

## BERT P2\_Fun Facts about BERT

### Why does BERT work?

"为什么BERT有用?"

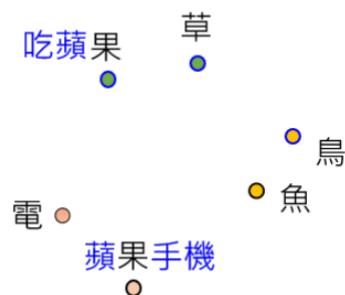
最常见的解释是, 当输入一串文本时, 每个文本都有一个对应的向量。对于这个向量, 我们称之为embedding。



它的特别之处在于，这些向量代表了输入词的含义。例如，模型输入 "台湾大学"（国立台湾大学），输出4个向量。这4个向量分别代表 "台"、"湾"、"大"和"学"

更具体地说，如果你把这些词所对应的向量画出来，或者计算它们之间的距离

The tokens with similar meaning have similar embedding.



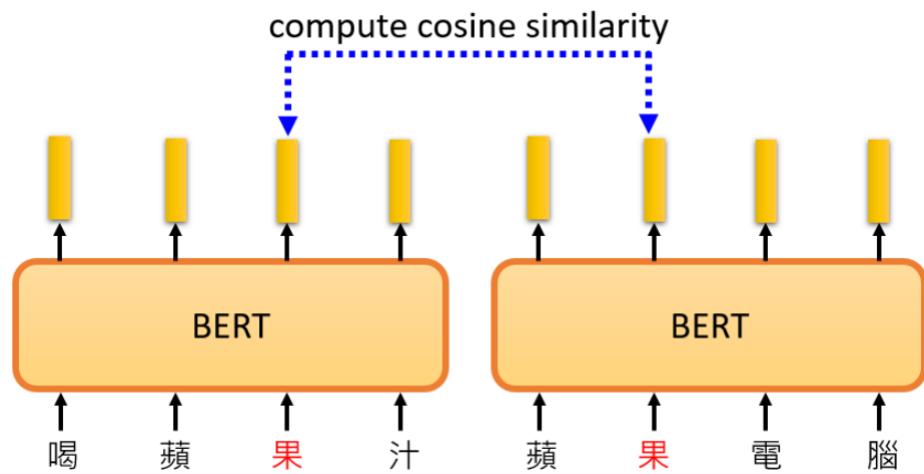
**Context is considered.**

你会发现，意思比较相似的词，它们的向量比较接近。例如，水果和草都是植物，它们的向量比较接近。但这是一个假的例子，我以后会给你看一个真正的例子。“鸟”和“鱼”是动物，所以它们可能更接近。

你可能会问，中文有歧义，其实不仅是中文，很多语言都有歧义，BERT可以考虑上下文，所以，同一个词，比如说“苹果”，它的上下文和另一个“苹果”不同，它们的向量也不会相同。

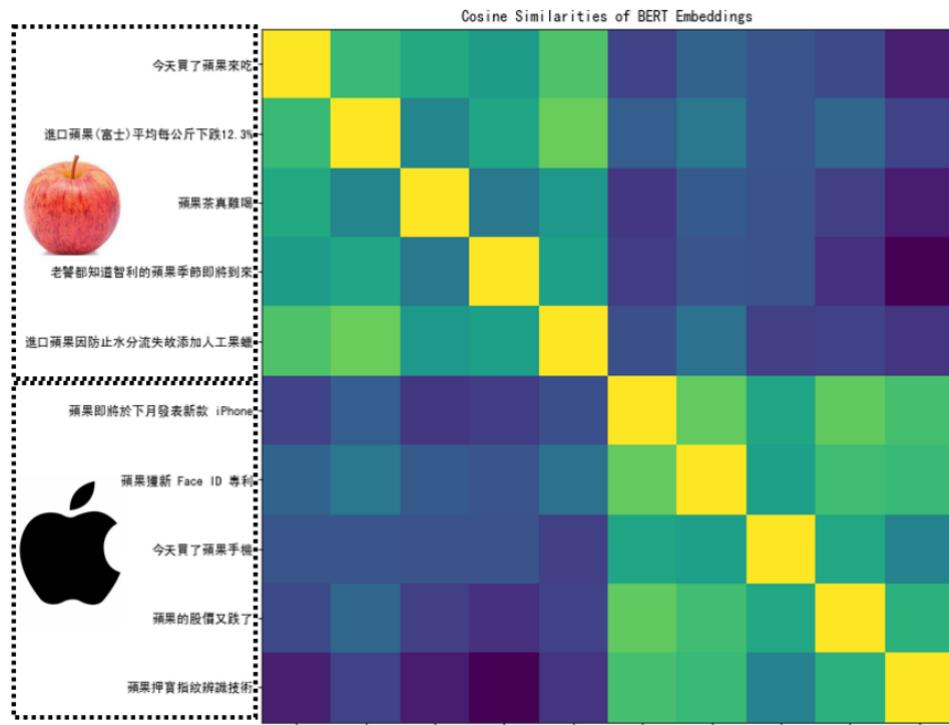
水果“苹果”和手机“苹果”都是“苹果”，但根据上下文，它们的含义是不同的。所以，它的向量和相应的embedding会有很大不同。水果“苹果”可能更接近于“草”，手机“苹果”可能更接近于“电”。

现在我们看一个真实的例子。假设我们现在考虑“苹果”这个词，我们会收集很多有“苹果”这个词的句子，比如“喝苹果汁”、“苹果Macbook”等等。然后，我们把这些句子放入BERT中。



接下来，我们将计算 "苹果"一词的相应embedding。输入 "喝苹果汁"，得到一个 "苹果" 的向量。为什么不一样呢？在Encoder中存在Self-Attention，所以根据 "苹果"一词的不同语境，得到的向量会有所不同。接下来，我们计算这些结果之间的cosine similarity，即计算它们的相似度。

结果是这样的，这里有10个句子



- 前5个句子中的 "苹果" 代表可食用的苹果。例如，第一句是 "我今天买了苹果吃"，第二句是 "进口富士苹果平均每公斤多少钱"，第三句是 "苹果茶很难喝"，第四句是 "智利苹果的季节来了"，第五句是 "关于进口苹果的事情"，这五个句子都有 "苹果" 一词，
- 后面五个句子也有 "苹果" 一词，但提到的是苹果公司的苹果。例如，"苹果即将在下个月发布新款 iPhone"，"苹果获得新专利"，"我今天买了一部苹果手机"，"苹果股价下跌"，"苹果押注指纹识别技术"，共有十个 "苹果"

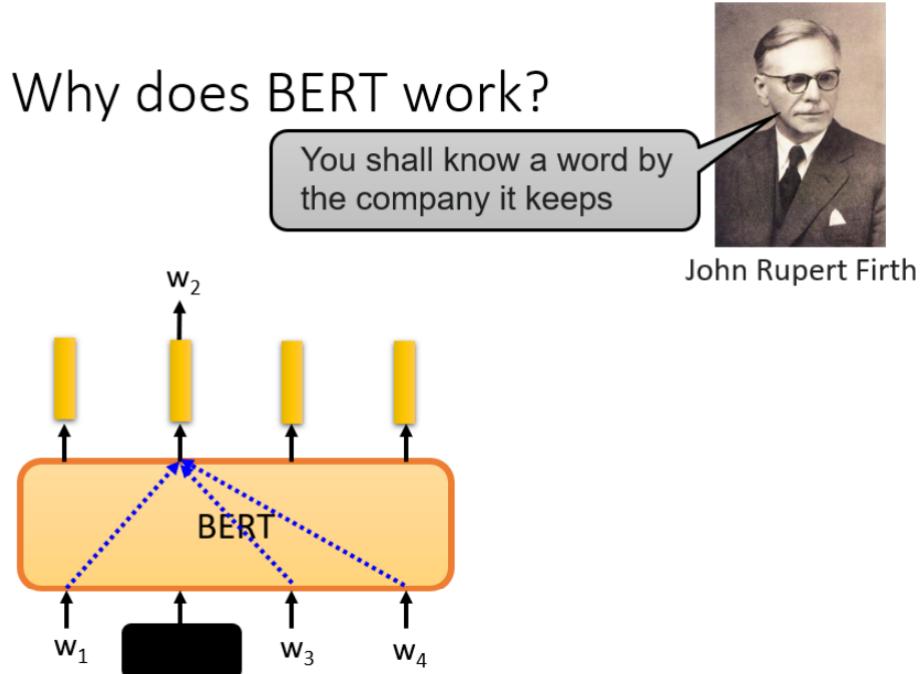
计算每一对之间的相似度，得到一个 $10 \times 10$ 的矩阵。相似度越高，这个颜色就越浅。所以，自己和自己之间的相似度一定是最高的，自己和别人之间的相似度一定是较小的。

但前五个 "苹果" 和后五个 "苹果" 之间的相似度相对较低。

BERT知道，前五个 "苹果" 是指可食用的苹果，所以它们比较接近。最后五个 "苹果" 指的是苹果公司，所以它们比较接近。所以BERT知道，上下两堆 "苹果" 的含义不同。

BERT的这些向量是输出向量，每个向量代表该词的含义。BERT在填空的过程中已经学会了每个汉字的意思。",也许它真的理解了中文，对它来说，汉字不再是毫无关联的，既然它理解了中文，它就可以在接下来的任务中做得更好。

那么接下来你可能会问，"为什么BERT有如此神奇的能力？",为什么.....,为什么它能输出代表输入词含义的向量？这里，约翰·鲁伯特·弗斯，一位60年代的语言学家，提出了一个假说。他说，要知道一个词的意思，我们需要看它的 "Company"，也就是经常和它一起出现的词汇，也就是它的上下文。



一个词的意思，取决于它的上下文

- 所以以苹果 (apple) 中的果字为例。如果它经常与 "吃"、"树"等一起出现，那么它可能指的是可食用的苹果。
- 如果它经常与电子、专利、股票价格等一起出现，那么它可能指的是苹果公司。

当我们训练BERT时，我们给它 $w_1$ 、 $w_2$ 、 $w_3$ 和 $w_4$ ，我们覆盖 $w_2$ ，并告诉它预测 $w_2$ ，而它就是从上下文中提取信息来预测 $w_2$ 。所以这个向量是其上下文信息的精华，可以用来预测 $w_2$ 是什么。

这样的想法在BERT之前已经存在了。在**word embedding**中，有一种技术叫做**CBOW**。

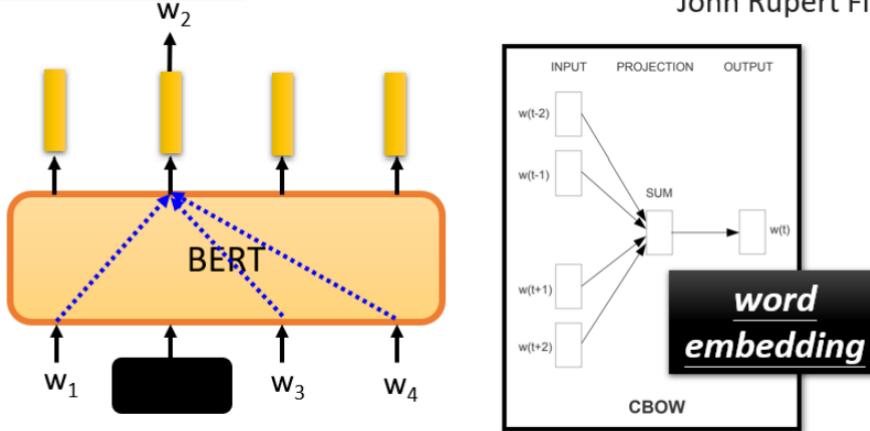
# Why does BERT work?



**Contextualized word embedding**

You shall know a word by the company it keeps

John Rupert Firth



CBOW所做的，与BERT完全一样。做一个空白，并要求它预测空白处的内容。这个CBOW，这个word embedding技术，可以给每个词汇一个向量，代表这个词汇的意义。

CBOW是一个非常简单的模型，它使用两个变换，是一个非常简单的模型，有人会问，“为什么它只使用两个变换？”，“它可以更复杂吗？”，CBOW的作者，Thomas Mikolov，曾经来到台湾。当时我在上课的时候，经常有人问我，为什么CBOW只用线性，为什么不用深度学习，我问过Thomas Mikolov这个问题，他说可以用深度学习，但是之所以选择线性模型，一个简单的模型，最大的担心，其实是算力问题。当时的计算能力和现在不在一个数量级上，可能是2016年的时候，几年前的技术也不在一个数量级上，当时要训练一个非常大的模型还是比较困难的，所以他选择了一个比较简单的模型。

今天，当你使用BERT的时候，就相当于一个深度版本的CBOW，你可以做更复杂的事情，而且BERT还可以根据不同的语境，从同一个词汇产生不同的embedding。因为它是一个考虑到语境的高级版本的词embedding，BERT也被称为Contextualized embedding，这些由BERT提取的向量或embedding被称为Contextualized embedding，希望大家能接受这个答案。

但是，这个答案，它真的是真的吗？这是你在文献中听到最多的答案。当你和别人讨论BERT时，这是大多数人都会告诉你的理由。它真的是真的吗？这里有一个难以理解的，由我们实验室的一个学生做的实验。实验是这样的：我们应用为文本训练的BERT对蛋白质、DNA链和音乐进行分类。

<https://arxiv.org/abs/2103.07162>  
This work is done by 高璋聰

## Why does BERT work?

- Applying BERT to protein, DNA, music classification



A  
T  
C  
G

class

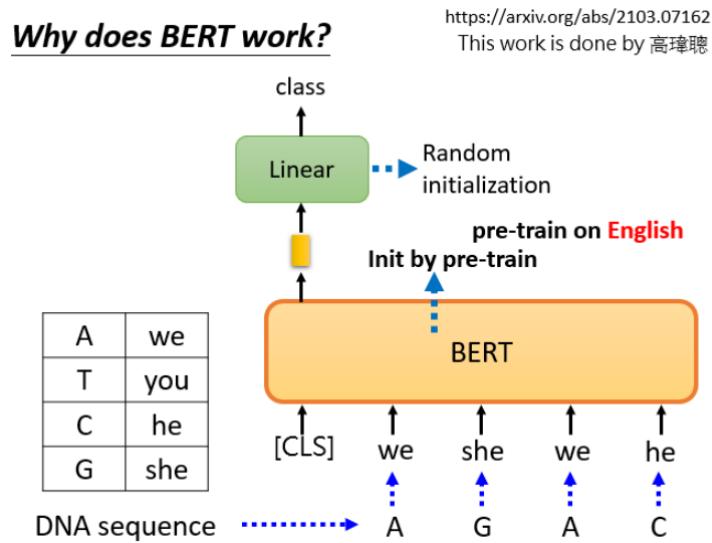
EI	CCAGCTGCATCACAGGAGGCCAGCC
EI	AGACCCGCCGGGAGGCAGGAGGACC
IE	AACGTGGCCTCCTTGCCCTTCCCC
IE	CCACTCAGCCAGGCCCTTCTTCCT
IE	CCTGATCTGGTCTCCCCTCCCACCC
IE	AGCCCTCAACCCCTCTGTCTCACCC
IE	CCACTCAGCCAGGCCCTTCTTCCT
N	CTGTGTTACCAACATCAAGCGCCGG
N	GTGTTACCGAGGGCATTCTAACAG
N	TCTGAGCTCTGCATTGTCTATTCTCC

DNA sequence

让我们以DNA链的分类为例。DNA是一系列的脱氧核苷酸，有四种，分别用A、T、C和G表示，所以一条DNA链是这样的。

你可能会问，“EI IE和N代表什么？”“不要在意细节，我也不知道，总之，这是一个分类问题。只要用训练数据和标记数据来训练它，就可以了。”

神奇的部分来了，DNA可以用ATCG来表示，现在，我们要用BERT来对DNA进行分类



例如，“A”是“we”，“T”是“you”，“C”是“he”，“G”是“she”。对应的词并不重要，你可以随机生成。“A”可以对应任何词汇，“T”、“C”和“G”也可以，这并不重要，对结果影响很小。只是这串文字无法理解。

例如，“AGAC”变成了“we she we he”，不知道它在说什么。

然后，把它扔进一个一般的BERT，用CLS标记，一个输出向量，一个Linear transform，对它进行分类。只是分类到了DNA类，我不知道他们是什么意思。

和以前一样，Linear transform使用随机初始化，而BERT是通过预训练模型初始化的。但用于初始化的模型，是学习填空的模型。它已经学会了英语填空。

你可能会认为，这个实验完全是无稽之谈。如果我们把一个DNA序列预处理成一个无意义的序列，那么BERT的目的是什么？大家都知道，BERT可以分析一个有效句子的语义，你怎么能给它一个无法理解的句子呢？做这个实验的意义是什么？

蛋白质有三种分类，那么蛋白质是由氨基酸组成的，有十种氨基酸，只要给每个氨基酸一个随机的词汇，那么DNA是一组ATCG，音乐也是一组音符，给它每个音符一个词汇，然后，把它作为一个文章分类问题来做。

## • Applying BERT to protein, DNA, music classification

	Protein			DNA				Music
	localization	stability	fluorescence	H3	H4	H3K9ac	Splice	composer
specific	69.0	76.0	63.0	87.3	87.3	79.1	94.1	-
BERT	64.8	74.5	63.7	83.0	86.2	78.3	97.5	55.2
re-emb	63.3	75.4	37.3	78.5	83.7	76.3	95.6	55.2
rand	58.6	65.8	27.5	75.6	66.5	72.8	95	36



你会发现，如果你不使用BERT，你得到的结果是蓝色部分，如果你使用BERT，你得到的结果是红色部分，这实际上更好，你们大多数人现在一定很困惑。

这个实验只能用神奇来形容，没有人知道它为什么有效，而且目前还没有很好的解释，我之所以要谈这个实验，是想告诉你们，要了解BERT的力量，还有很多工作要做。

我并不是要否认BERT能够分析句子的含义这一事实。从embedding中，我们清楚地观察到，BERT知道每个词的含义，它能找出含义相似的词和不相似的词。但正如我想指出的那样，即使你给它一个无意义的句子，它仍然可以很好地对句子进行分类。

所以，也许它的力量并不完全来自于对实际文章的理解。也许还有其他原因。例如，也许，BERT只是一套更好的初始参数。也许这与语义不一定有关。也许这套初始参数，只是在训练大型模型时更好。

是这样吗？**这个问题需要进一步研究来回答。**我之所以要讲这个实验，是想让大家知道，我们目前使用的模型往往是非常新的，需要进一步的研究，以便我们了解它的能力。

你今天学到的关于BERT的知识，只是沧海一粟。我会把一些视频的链接放在这里。

To Learn More .....

**BERT (Part 1)**



[https://youtu.be/1\\_gRK9ElQpc](https://youtu.be/1_gRK9ElQpc)

**BERT (Part 2)**

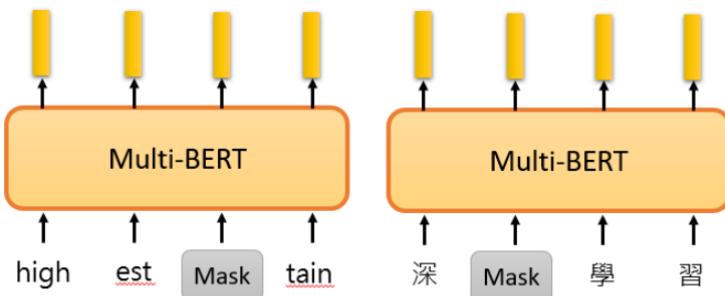


<https://youtu.be/Bywo7m6ySlk>

如果你想了解更多关于BERT的知识，你可以参考这些链接。你的作业不需要它，这学期剩下的时间也不需要。我只想告诉你，BERT还有很多其他的变种。

## Multi-lingual BERT

接下来，我要讲的是，一种叫做Multi-lingual BERT的BERT。Multi-lingual BERT有什么神奇之处？

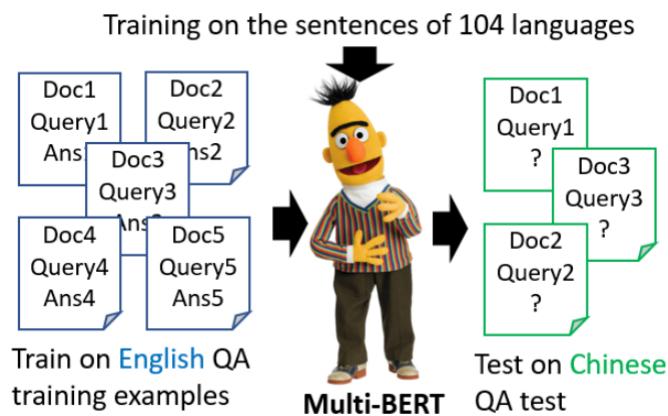


Training a BERT model by many different languages.

它是由很多语言来训练的，比如中文、英文、德文、法文等等，用填空题来训练BERT，这就是Multi-lingual BERT的训练方式。

## Zero-shot Reading Comprehension

google训练了一个Multi-lingual BERT，它能够做这104种语言的填空题。神奇的地方来了，如果你用英文问答数据训练它，它就会自动学习如何做中文问答



我不知道你是否完全理解我的意思，所以这里有一个真实的实验例子。

- English: SQuAD, Chinese: DRCD

Model	Pre-train	Fine-tune	Test	EM	F1
BERT	none	Chinese	Chinese	66.1	78.1
		Chinese		82.0	89.1
	104 languages	Chinese	Chinese	81.2	88.7
		English		63.3	78.8
			Chinese + English	82.6	90.1

F1 score of Human performance is 93.30%

This work is done by 劉記良、許宗嫄  
<https://arxiv.org/abs/1909.09587>

这是一些训练数据。他们用SQuAD进行fine-tune。这是一个英文Q&A数据集。中文数据集是由台达电发布的，叫DRCD。这个数据集也是我们在作业中要用到的数据集。

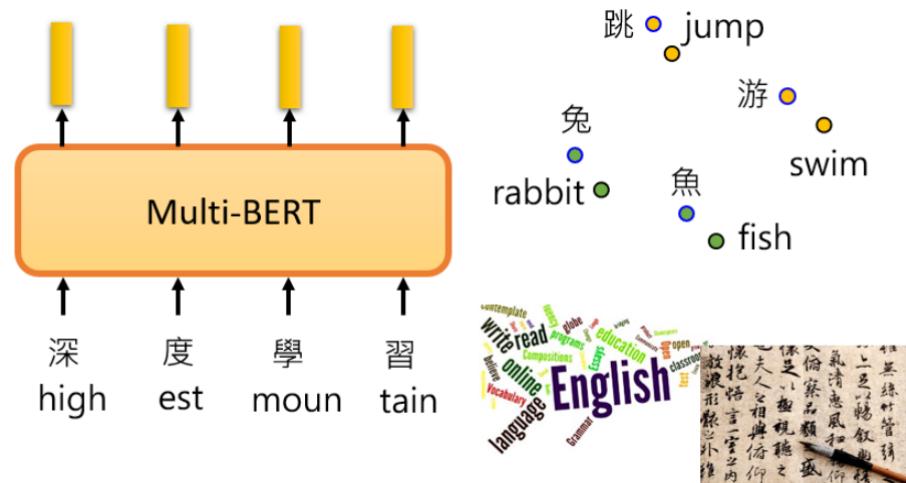
在BERT提出之前，效果并不好。在BERT之前，最强的模型是QANet。它的正确率只有……，嗯，我是说F1得分，而不是准确率，但你可以暂时把它看成是准确率或正确率。

如果我们允许用中文填空题进行预训练，然后用中文Q&A数据进行微调，那么它在中文Q&A测试集上的正确率达到89%。因此，其表现是相当令人印象深刻的。

神奇的是，如果我们把一个Multi-lingual的BERT，用英文Q&A数据进行微调，它仍然可以回答中文Q&A问题，并且有78%的正确率，这几乎与QANet的准确性相同。它从未接受过中文和英文之间的翻译训练，也从未阅读过中文Q&A的数据收集。,它在没有任何准备的情况下参加了这个中文Q&A测试，尽管它从未见过中文测试，但不知为何，它能回答这些问题。

## Cross-lingual Alignment?

你们中的一些人可能会说：“它在预训练中读过104种语言，104种语言中的一种是中文，是吗？如果是，这并不奇怪。”但是在预训练中，学习的目标是填空。它只能用中文填空。有了这些知识，再加上做英文问答的能力，不知不觉中，它就自动学会了做中文问答。

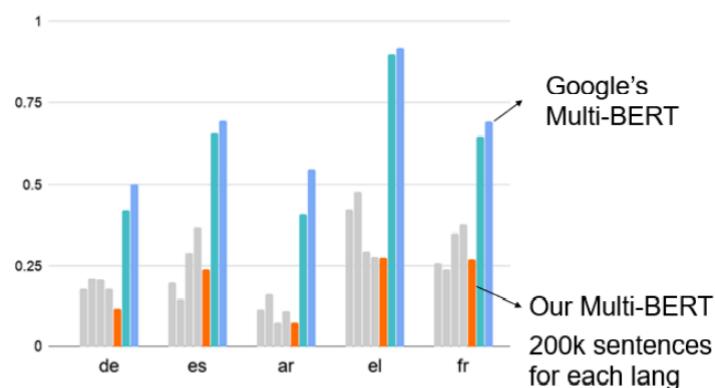


听起来很神奇，那么BERT是怎么做到的呢？一个简单的解释是：也许对于多语言的BERT来说，不同的语言并没有那么大的差异。无论你用中文还是英文显示，对于具有相同含义的单词，它们的embedding都很接近。汉语中的“跳”与英语中的“jump”接近，汉语中的“鱼”与英语中的“fish”接近，汉语中的“游”与英语中的“swim”接近，也许在学习过程中它已经自动学会了。

它是可以被验证的。我们实际上做了一些验证。验证的标准被称为Mean Reciprocal Rank，缩写为MRR。我们在这里不做详细说明。你只需要知道，MRR的值越高，不同embedding之间的Alignment就越好。

更好的Alignment意味着，具有相同含义但来自不同语言的词将被转化为更接近的向量。**如果MRR高，那么具有相同含义但来自不同语言的词的向量就更接近。**

Mean Reciprocal Rank (MRR):  
Higher MRR, better alignment



<https://arxiv.org/abs/2010.10938>  
投影片來源: 許宗婷同學碩士口試投影片

How about 1000k?

这条深蓝色的线是谷歌发布的104种语言的Multi-lingual BERT的MRR，它的值非常高，这说明不同语言之间没有太大的差别。Multi-lingual BERT只看意思，不同语言对它没有太大的差别。

橙色这条是我们试图自己训练Multi-lingual BERT。我们使用的数据较少，每种语言只使用了20万个句子。数据较少。我们自我训练的模型结果并不好。我们不知道为什么我们的Multi-lingual BERT不能将不同的语言统一起来。似乎它不能学习那些在不同语言中具有相同含义的符号，它们应该具有相同的含义。这个问题困扰了我们很长时间。

为什么我们要做这个实验？为什么我们要自己训练Multi-lingual BERT？因为我们想了解，是什么让Multi-lingual BERT。我们想设置不同的参数，不同的向量，看看哪个向量会影响Multi-lingual BERT。

但是我们发现，对于我们的Multi-lingual BERT来说，无论你如何调整参数，它就是不能达到Multi-lingual的效果，它就是不能达到Alignment的效果。我们把数据量增加了五倍，看看能不能达到Alignment的效果。在做这个实验之前，大家都有点抵触，大家都觉得有点害怕，因为训练时间要比原来的长五倍。

训练了两天后，什么也没发生，损失甚至不能减少，就在我们要放弃的时候，损失突然下降了

## The training is also challenging ...



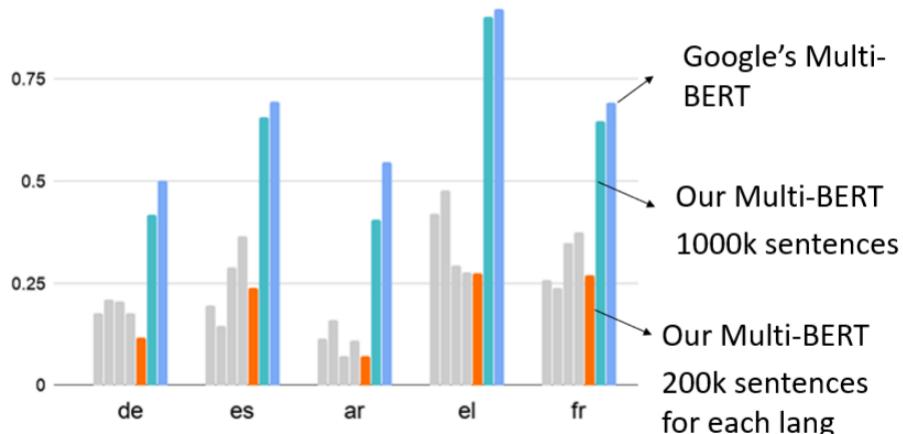
用了8个V100来训练，我们的实验室也没有8个V100，是在NCHC（国家高性能计算中心）的机器上运行的，训练了两天后，损失没有下降，似乎失败了。当我们要放弃的时候，损失下降了。

这是某个学生在Facebook上发的帖子，我在这里引用它来告诉你，我当时心里的感叹。整个实验，必须运行一个多星期，才能把它学好，每一种语言1000K的数据。

## Mean Reciprocal Rank (MRR):

Higher MRR, better alignment

1 — The amount of training data is critical for alignment.



<https://arxiv.org/abs/2010.10938>

投影片來源: 許宗嫄同學碩士口試投影片

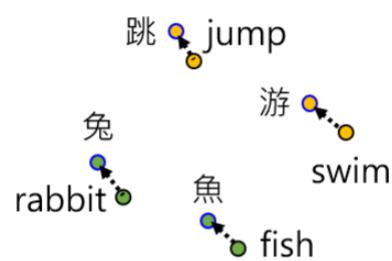
所以看起来，数据量是一个非常关键的因素，关系到能否成功地将不同的语言排列在一起。所以有时候，神奇的是，很多问题或很多现象，只有在有足够的数据量时才会显现出来。它可以在A语言的QA上进行训练，然后直接转移到B语言上，从来没有人说过这一点

这是过去几年才出现的，一个可能的原因是，过去没有足够的数据，现在有足够的数据，现在有大量的计算资源，所以这个现象现在有可能被观察到。

最后一个神奇的实验，我觉得这件事很奇怪

<https://arxiv.org/abs/2010.10041>

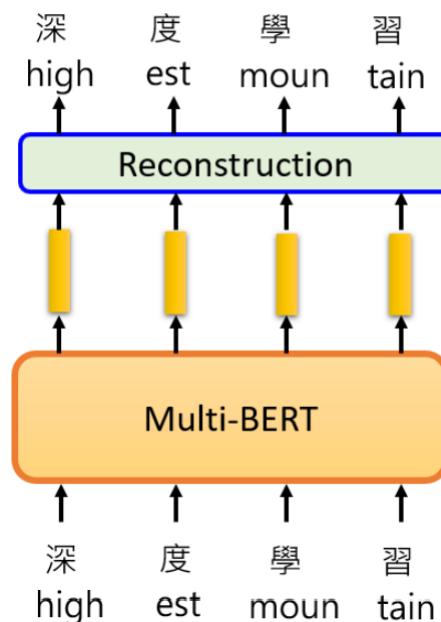
### Weird???



If the embedding is language independent ...

How to correctly reconstruct?

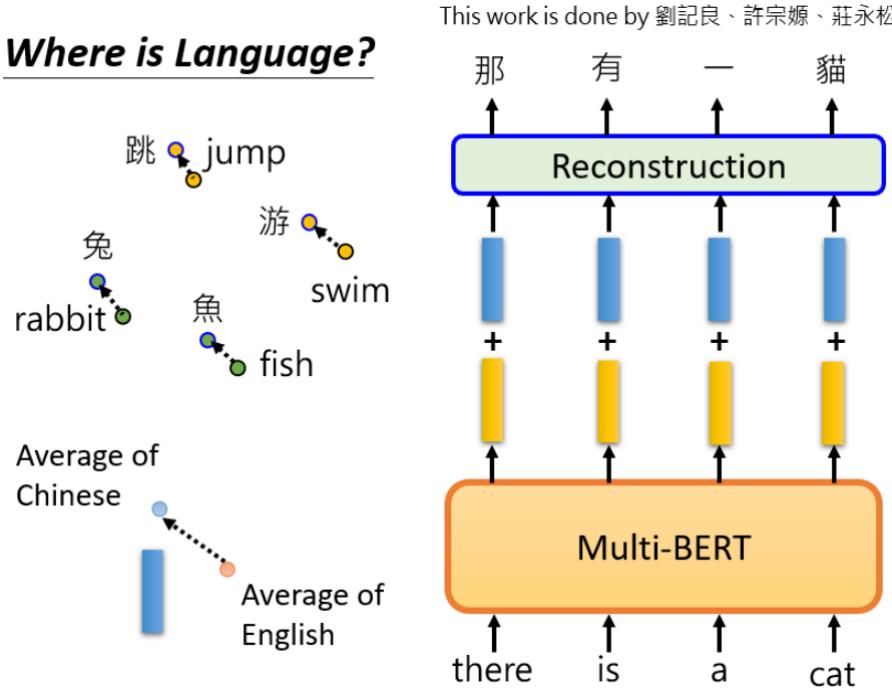
There must be language information.



你说BERT可以把不同语言中含义相同的符号放在一起，使它们的向量接近。但是，当训练多语言的BERT时，如果给它英语，它可以用英语填空，如果给它中文，它可以用中文填空，它不会混在一起

那么，如果不同语言之间没有区别，怎么可能只用英语标记来填英语句子呢？为什么它不会用中文符号填空呢？它就是不填，这说明它知道语言的信息也是不同的，那些不同语言的符号毕竟还是不同的，它并没有完全抹去语言信息，所以我想出了一个研究课题，我们来找找，语言信息在哪里。

后来我们发现，语言信息并没有隐藏得很深。一个学生发现，我们把所有英语单词的embedding，放到多语言的BERT中，取embedding的平均值，我们对中文单词也做同样的事情。在这里，我们给Multi-lingual BERT一个英语句子，并得到它的embedding。我们在embedding中加上这个蓝色的向量，这就是英语和汉语之间的差距。

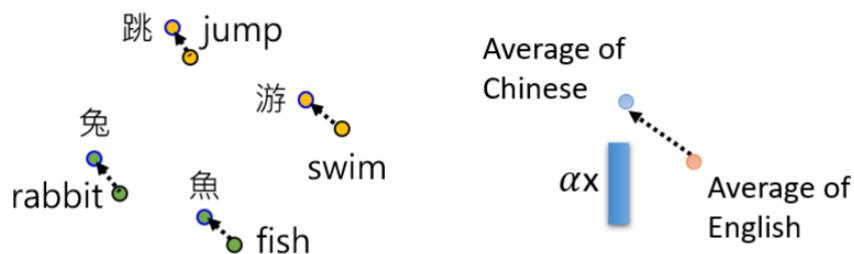


这些向量，从Multi-lingual BERT的角度来看，变成了汉语。有了这个神奇的东西，你可以做一个奇妙的无监督翻译。

例如，你给BERT看这个中文句子。

This work is done by 劉記良、許宗嫄、莊永松  
<https://arxiv.org/abs/2010.10041>

### If this is true ...



Input (en) | The girl that can help me is all the way across town. There is no one who can help me.

Ground Truth (zh)	能帮助我的女孩在小镇的另一边。没有人能帮助我。 en→zh, $\alpha = 1$ .孩, can 来我是all the way across 市。。There 是无人 can help 我。 en→zh, $\alpha = 2$ .孩的家我是这个人的市。。他是他人的到我。 en→zh, $\alpha = 3$ 。, 的的的他是的个的, 。: 他是他人, 的。他。
-------------------	--

Unsupervised token-level translation ☺

这个中文句子是，“能帮助我的小女孩在小镇的另一边，，没人能够帮助我”，现在我们把这个句子扔到Multi-lingual BERT中。

然后我们取出Multi-lingual BERT中的一个层，它不需要是最最后一层，可以是任何一层。我们拿出某一层，给它一个embedding，加上这个蓝色的向量。对它来说，这个句子马上就从中文变成了英文。

在向BERT输入英文后，通过在中间加一个蓝色的向量来转换隐藏层，转眼间，中文就出来了。“没有人可以帮助我”，变成了“是（是）没有人（没有人）可以帮助我（我）”，“我”变成了“我”，“没有人”变成了“没有人”，所以它在某种程度上可以做无监督的标记级翻译，尽管它并不完美，神奇的是，Multilingual的BERT仍然保留了语义信息。

## BERT P3\_GPT3

除了BERT以外，还有下一个，也是鼎鼎有名的模型，就是GPT系列的模型



BERT做的是填空题，GPT就是改一下我们现在在self-supervised learning的时候，要模型做的任务

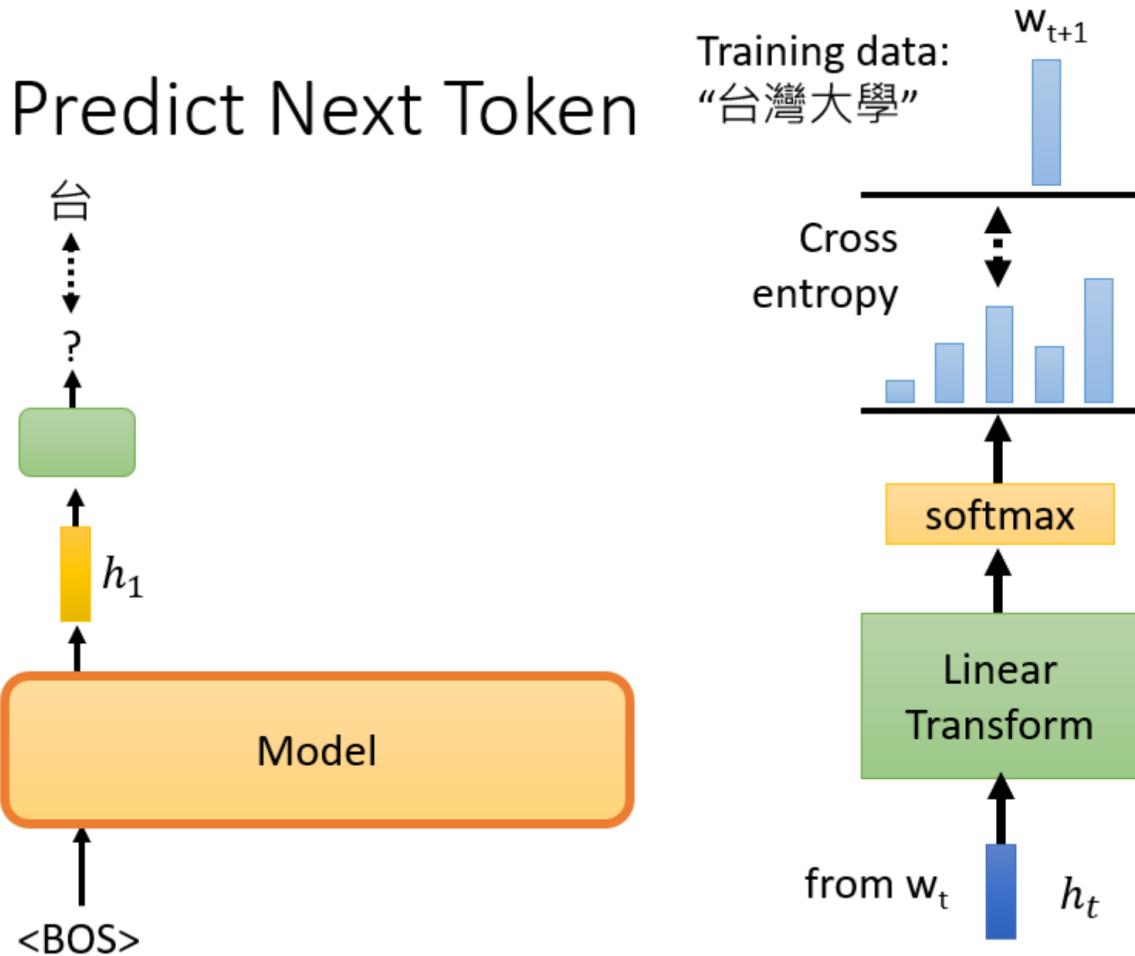
### Pre-train GPT——Predict Next Token

GPT要做的任务是，预测接下来，会出现的token是什么

举例来说，假设你的训练资料裡面，有一个句子是台湾大学，那GPT拿到这一笔训练资料的时候，它做的事情是这样

你给它BOS这个token，然后GPT output一个embedding，然后接下来，你用这个embedding去预测下一个，应该出现的token是什么

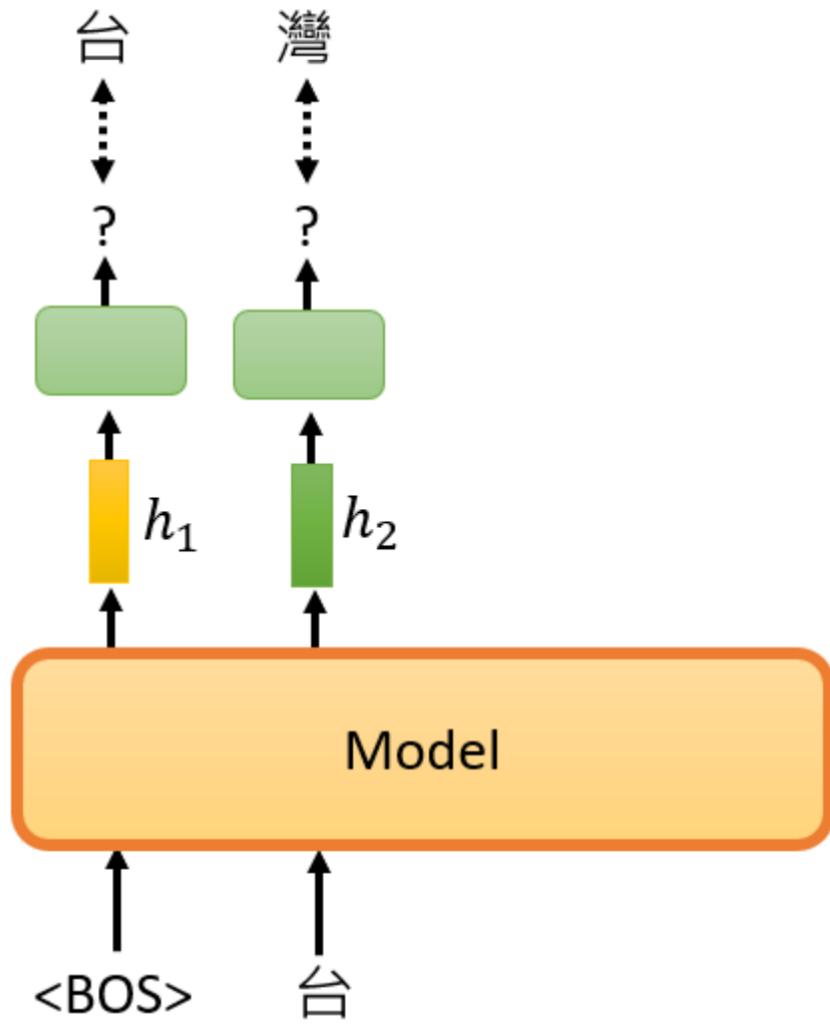
# Predict Next Token



那在这个句子裡面,根据这笔训练资料,下一个应该出现的token是"台",所以你要训练你的模型,根据第一个token,根据BOS给你的embedding,那它要输出"台"这个token

这个部分,详细来看就是这样,你有一个embedding,这边用 $h$ 来表示,然后通过一个Linear Transform,再通过一个softmax,得到一个distribution,跟一般你做分类的问题是一样的,接下来,你希望你output的distribution,跟正确答案的Cross entropy,越小越好,也就是你要去预测,下一个出现的token是什麼

好那接下来要做的事情,就是以此类推了,你给你的GPT,"BOS"跟"台",它產生embedding,接下来它会预测,下一个出现的token是什麼,那你告诉它说,下一个应该出现的token,是"湾"



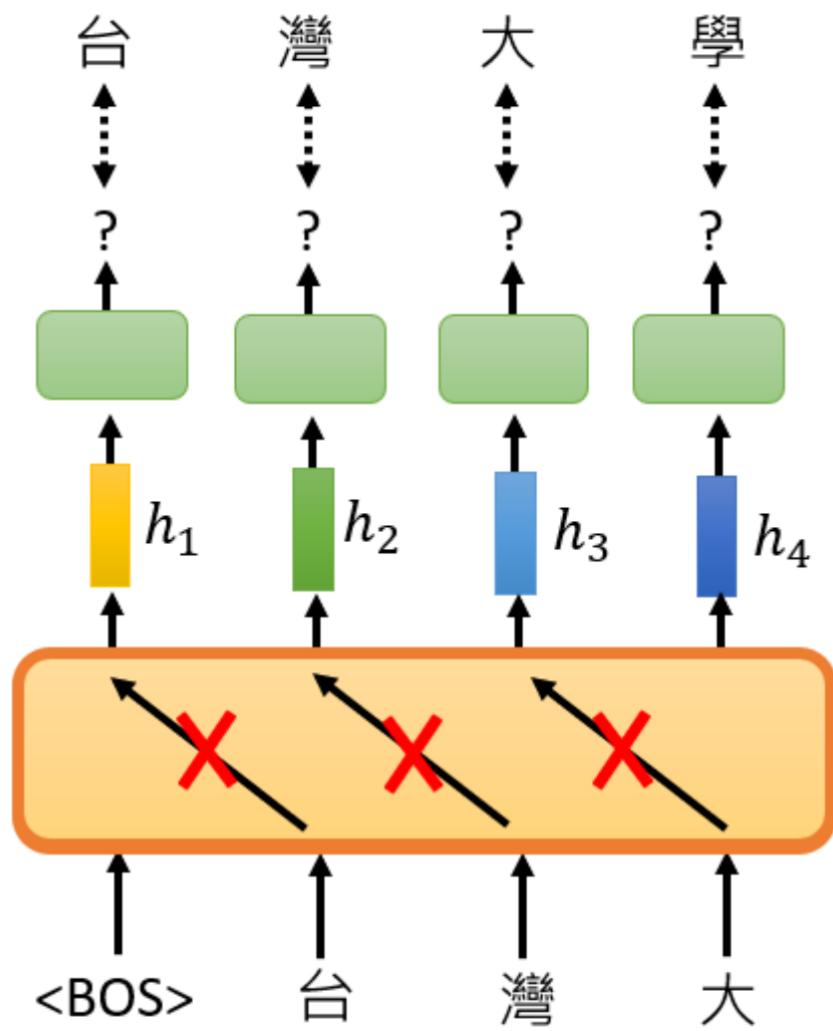
好 再反覆继续下去,你给它BOS "台"跟"湾",然后预测下一个应该出现的token,它应该要预测"大"

你给它"台"跟"湾"跟"大",接下来,下一个应该出现的token是"学"

那这边呢,是指拿一笔资料 一个句子,来给GPT训练,当然实际上你不会只用一笔句子,你会用成千上万个句子,来训练这个模型,然后就这样子说完了

它厉害的地方就是,用了很多资料,训了一个异常巨大的模型

那这边有一个小小的,应该要跟大家说的地方,是说这个GPT的模型,它像是一个transformer的decoder,不过拿掉BOS的attention这个部分,也就是说,你会做那个mask的attention



就是你现在在预测给BOS,预测台的时候,你不会看到接下来出现的词汇,给它台要预测湾的时候,你不会看到接下来要输入的词汇,以此类推 这个就是GPT

那这个GPT最知名的就是,因为**GPT可以预测下一个token,那所以它有生成的能力**,你可以让它不断地预测下一个token,产生完整的文章,所以我每次提到GPT的时候,它的形象都是一只独角兽

# Predict Next Token

They can do generation.



(PROMPT WRITTEN)

In a shocking finding, scientist discovered a herd of unicorns living in a remote, previously unexplored valley, in the Andes Mountains. Even more surprising to the researchers was the fact that the unicorns spoke perfect English.

(MODEL COMPLETION MACHINE-10 TRIES)

The scientist named the population, after their distinctive horn, Ovid's Unicorn. These four-horned, silver-white unicorns were previously unknown to science.

Now, after almost two centuries, the mystery of what sparked this odd phenomenon is finally solved.

Dr. Jorge Pérez, an evolutionary biologist from the University of La Paz, and several companions, were exploring the Andes Mountains when they found a small valley, with no other animals or humans. Pérez noticed that the valley had what appeared to be a natural fountain, surrounded by two peaks of rock and silver snow.

GPT系列最知名的一个例子,就是用GPT写了一篇,跟独角兽有关的新闻,因為他放一个假新闻,然后那个假新闻裡面说,在安地斯山脉发现独角兽等等,一个活灵活现的假新闻

為了让你更清楚了解,GPT运作起来是什麼样子,那这个线上有一个demo的网页,叫做talk to transformer,就是有人把一个比较小的,不是那个最大的GPT的模型,不是public available的,有人把比较小的GPT模型放在线上,让你可以输入一个句子,让它会把接下来的其餘的内容,把它补完

## How to use GPT?

怎麼把它用在downstream 的任务上呢,举例来说,怎麼把它用在question answering,或者是其他的,跟人类语言处理有关的任务上呢

**GPT用的想法跟BERT不一样,其实我要强调一下,GPT也可以跟BERT用一样的做法**

在使用BERT时, 把BERT model 拿出来,后面接一个简单的linear的classifier,那你就可以做很多事情,你也可以把GPT拿出来,接一个简单的classifier,我相信也是会有效

但是在GPT的论文中,它没有这样做,它有一个更狂的想法,為什麼会有更狂的想法呢,因為首先就是,BERT那一招BERT用过了嘛,所以总不能再用一样的东西,这样写paper就没有人觉得厉害了,然后再来就是,GPT这个模型,也许真的太大了,大到连fine tune可能都有困难

我们在用BERT的时候,你要把BERT模型,后面接一个linear classifier,然后BERT也是你的,要train的model的一部分,所以它的参数也是要调的,所以在刚才助教公告的,BERT相关的作业裡面,你还是需要花一点时间来training,虽然助教说你大概20分鐘,就可以train完了,因為你并不是要train一个,完整的BERT的模型,BERT的模型在之前,在做这个填空题的时候,已经训练得差不多了,你只需要微调它就好了,但是微调还是要花时间的,也许GPT实在是太过巨大,巨大到要微调它,要train一个epoch,可能都有困难,所以GPT系列,有一个更狂的使用方式

这个更狂的使用方式和人类更接近,你想想看假设你去考,譬如说托福的听力测验,你是怎麼去考

## 第一部份：詞彙和結構

本部份共 15 題，每題含一個空格。請就試題冊上 A、B、C、D 四個選項中選出最適合題意的字或詞，標示在答案紙上。

例：

It's eight o'clock now. Sue \_\_\_\_\_ in her bedroom.

- A. study
- B. studies
- C. studied
- D. is studying

正確答案為 D，請在答案紙上塗黑作答。

## Description

### A few example

- 首先你会看到一个题目的说明,告诉你说现在要考选择题,请从ABCD四个选项裡面,选出正确的答案等等
- 然后给你一个范例,告诉你说这是题目,然后正确的答案是多少
- 然后你看到新的问题,期待你就可以举一反三开始作答

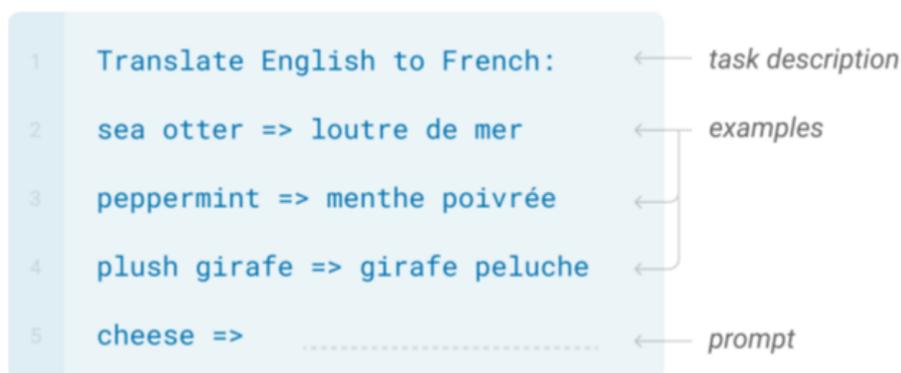
GPT系列要做的事情就是,这个模型能不能够,做一样的事情呢

## "In-context" Learning

## "Few-shot" Learning

举例来说假设要GPT这个模型做翻译

**"Few-shot"**  
**Learning**  
**(no gradient descent)**



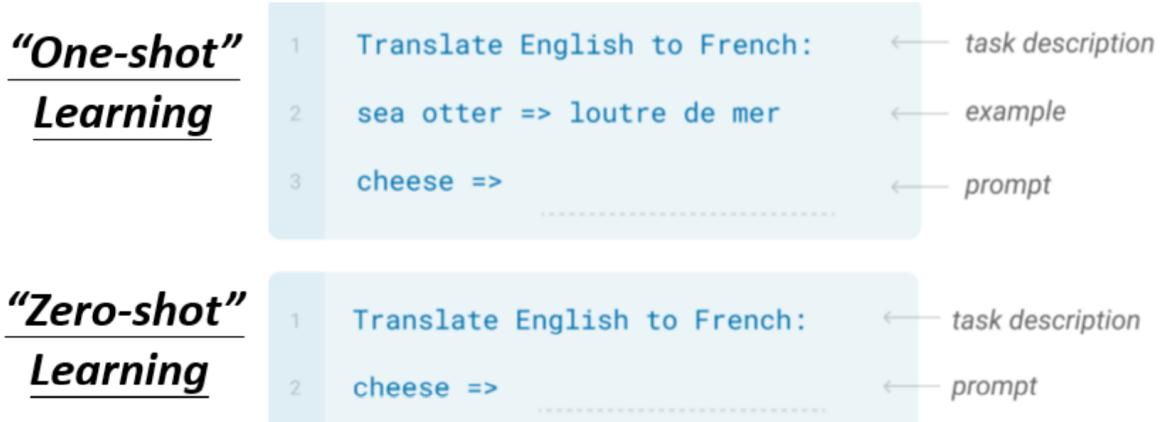
- 你就先打Translate English to French
- 就先给它这个句子,这个句子代表问题的描述
- 然后给它几个范例跟它说,sea otter然后=>,后面就应该长这个样子
- 或者是这个什麼plush girafe,plush girafe后面,就应该长这个样子等等
- 然后接下来,你问它说cheese=>,叫它把后面的补完,希望它就可以產生翻译的结果

不知道大家能不能够了解,这一个想法是多麼地狂,在training的时候,GPT并没有教它做翻译这件事,它唯一学到的就是,给一段文字的前半段,把后半段补完,就像我们刚才给大家示范的例子一样,现在我们直接给它前半段的文字,就长这个样子,告诉它说你要做翻译了,给你几个例子,告诉你说翻译是怎麼回事,接下来给它cheese这个英文单字,后面能不能就直接接出,法文的翻译结果呢

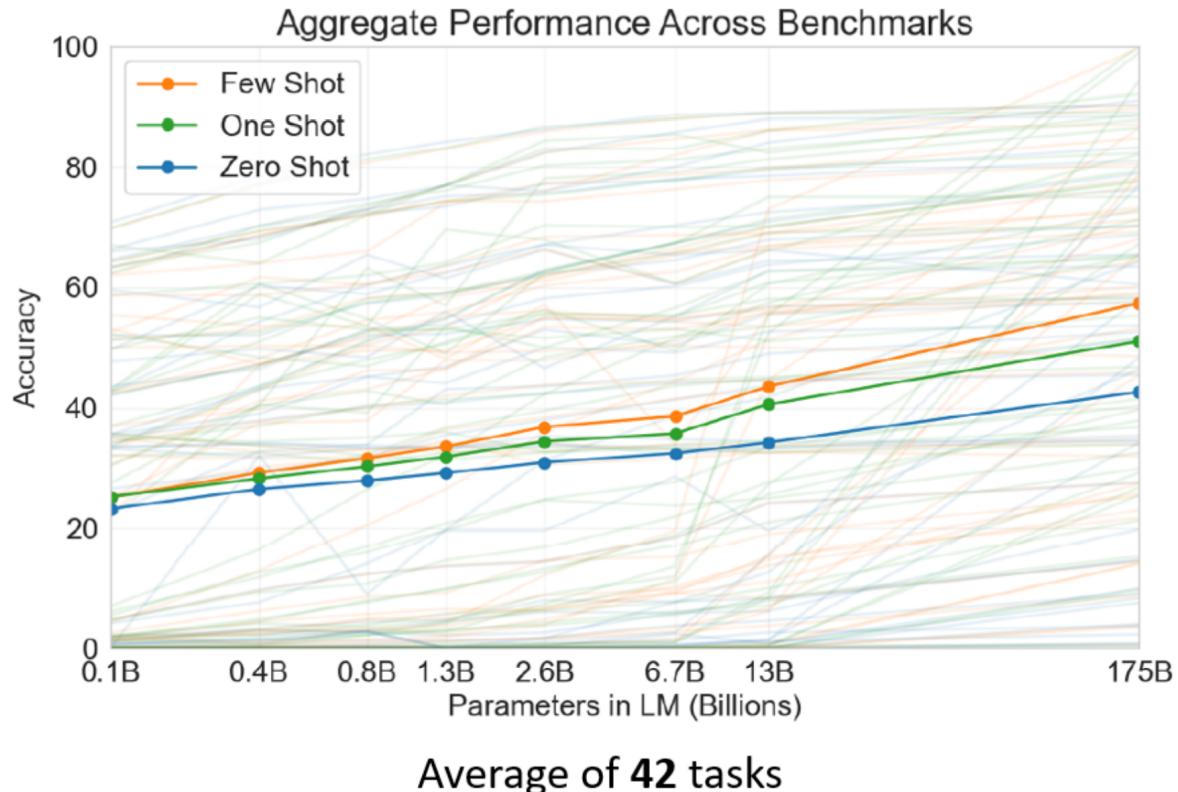
这个在GPT的文献裡面,叫做Few-shot Learning,但是它跟一般的Few-shot Learning,又不一样,所谓Few Shot的意思是说,确实只给了它一点例子,所以叫做Few Shot,但是它不是一般的learning,这裡面完全没有gradient descent,完全没有要去调,GPT那个模型参数的意思,所以在GPT的文献裡面,把这种训练给了一个特殊的名字,它们叫做In-context Learning,代表说它不是一种,一般的Few-shot learning,它连gradient descent都没有做

## “One-shot” Learning & “Zero-shot” Learning

当然你也可以给GPT更大的挑战,我们在考托福听力测验的时候,都只给一个例子而已,那GPT可不可以只看一个例子,就知道它要做翻译这件事,这个叫One-shot Learning



还有更狂的,是Zero-shot Learning,直接给它一个叙述,说我们现在要做翻译了,GPT能不能够自己就看得懂,就自动知道说要来做翻译这件事情呢,那如果能够做到的话,那真的就非常地惊人了,那GPT系列,到底有没有达成这个目标呢,这个是一个见仁见智的问题啦



它不是完全不可能答对,但是正确率有点低,相较於你可以微调模型,正确率是有点低的,那细节你就再看看GPT那篇文章

第三代的GPT,它测试了42个任务,这个纵轴是正确率,这些实线这三条实线,是42个任务的平均正确率,那这边包括了Few Shot,One Shot跟Zero Shot,三条线分别代表Few Shot,One Shot跟Zero Shot,横轴代表模型的大小,它们测试了一系列不同大小的模型,从只有0.1个billion的参数,到175个billion的参数,那从只有0.1个billion的参数,到175个billion的参数,我们看Few Shot的部分,从20几%的正确率 平均正确率,一直做到50几%的平均正确率,那至於50几%的平均正确率,算是有做起来 还是没有做起来,那这个就是见仁见智的问题啦

目前看起来状况是,有些任务它还真的学会了,举例来说2这个加减法,你给它一个数字加另外一个数字,它真的可以得到,正确的两个数字加起来的结果,但是有些任务,它可能怎麽学都学不会,譬如说一些跟逻辑推理有关的任务,它的结果就非常非常地惨,好 那有关GPT3的细节,这个就留给大家再自己研究,然后这边有一个过去上课的录影,我把连结放在这边给大家参考



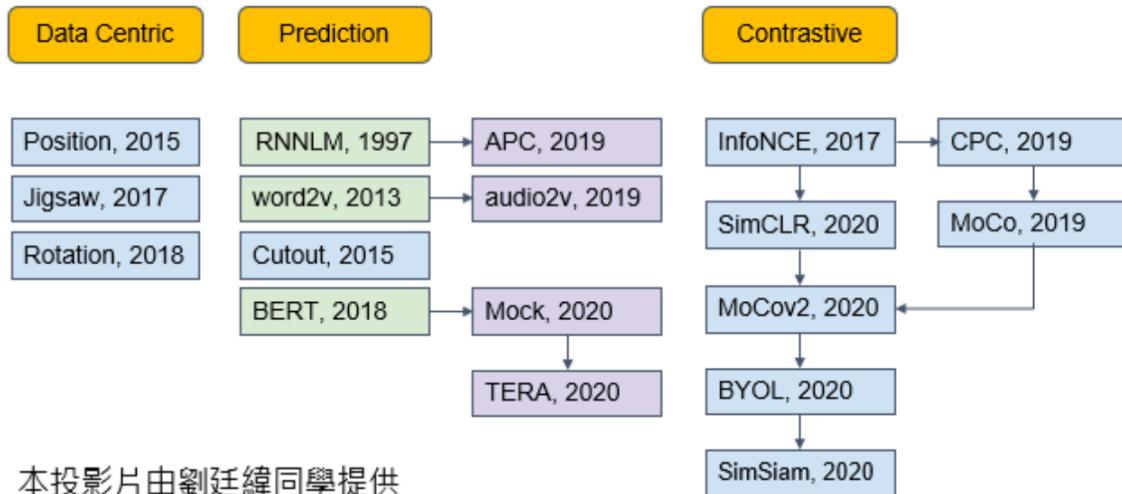
<https://youtu.be/DOG1L9lvsDY>

## Beyond Text

---

到目前为止我们举的例子,都是只有跟文字有关,但是你不要误会说,这种self-supervised learning的概念,只能用在文字上

在CV,CV就是computer vision,也就是影像,在语音跟影像的应用上也都可以用,self-supervised learning的技术,那其实今天,self-supervised learning的技术,非常非常多,我们讲的BERT跟GPT系列,它只是三个类型的,这个self-supervised learning的方法,的其中一种,它们是属于prediction那一类



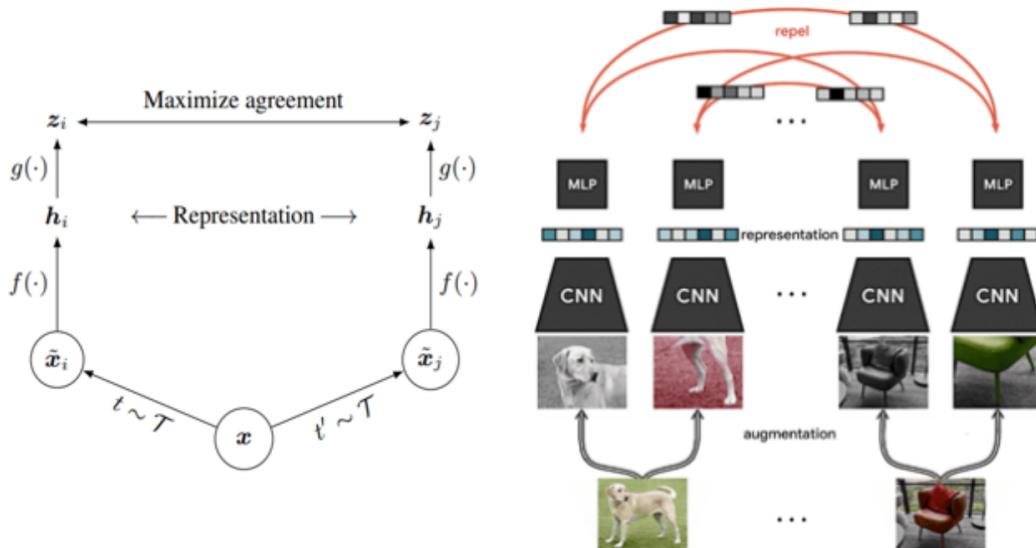
那其实还有其他的类型,那就不是我们这一堂课要讲的,那接下来的课程,你可能会觉得有点流水帐,就是我们每一个主题呢,就是告诉你说这个主题裡面,有什麼但是细节这个更多的知识,就留给大家自己来做更进一步的研究,所以这些投影片,只是要告诉你说,在self-supervised learning这个部分,我们讲的只是整个领域的其中一小块,那还有更多的内容,是等待大家去探索的

## Image - SimCLR

好那有关影像的部分呢,我们就真的不会细讲,我这边就是放两页投影片带过去,告诉你说有一招非常有名的,叫做SimCLR,它的概念也不难,我相信你自己读论文,应该也有办法看懂它

## Image - SimCLR

<https://arxiv.org/abs/2002.05709>  
<https://github.com/google-research/simclr>

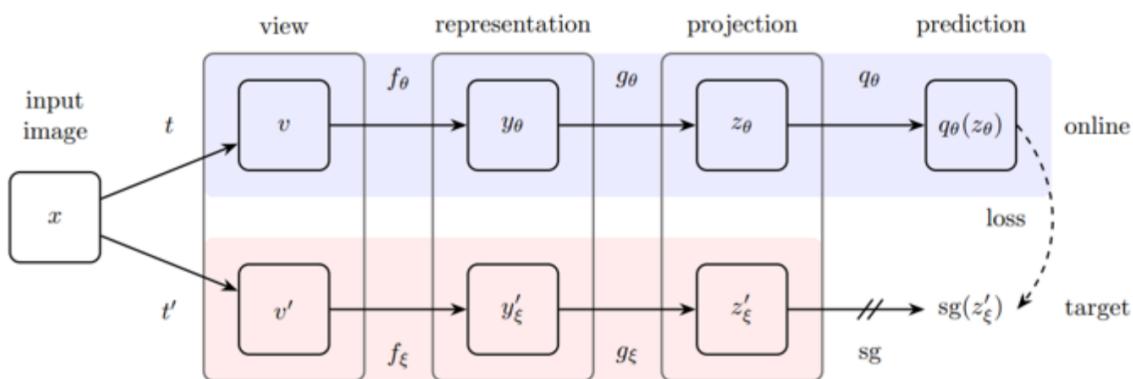


## Image - BYOL

那还有很奇怪的,叫做BYOL

# Bootstrap your own latent: A new approach to self-supervised Learning

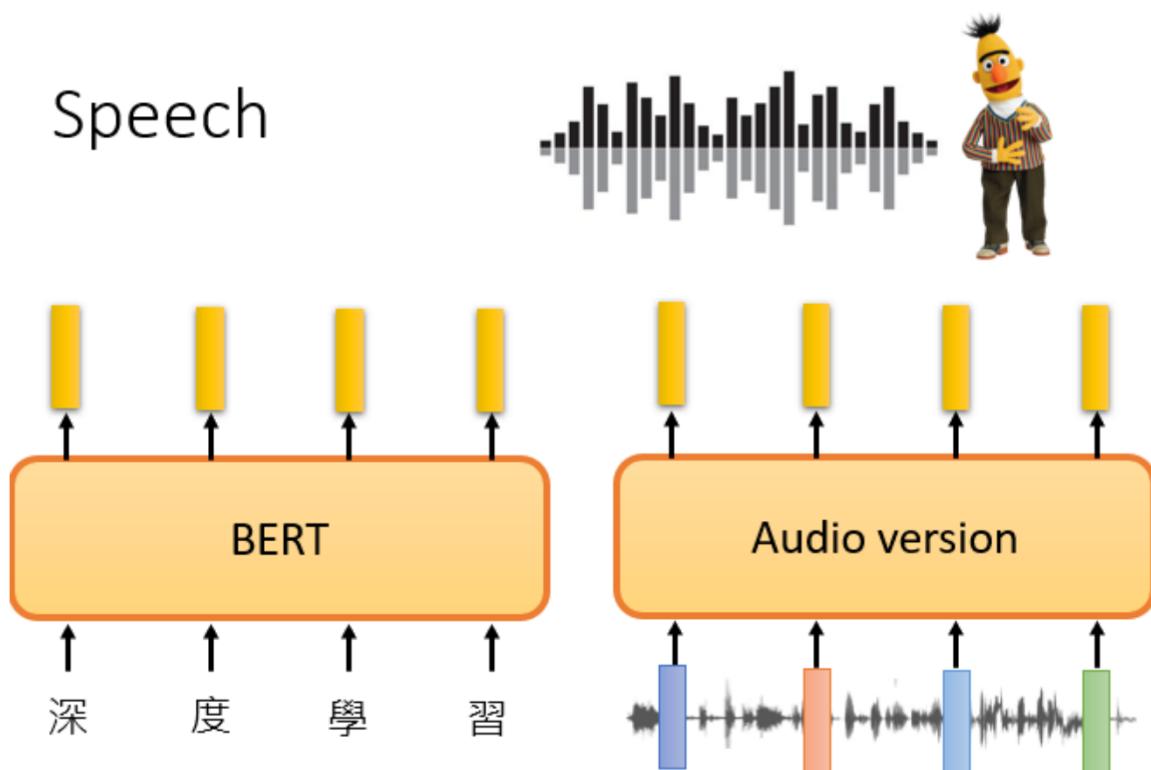
<https://arxiv.org/abs/2006.07733>



BYOL这个东西呢,我们是不太可能在上课讲它,為什麼呢,因為根本不知道它為什麼会work,不是这个是很新的论文,这个是去年夏天的论文,那这个论文是,假设它不是已经发表的文章,然后学生来跟我提这个想法,我一定就是,我一定不会让他做,这不可能会work的,这是个不可能会实现的想法,不可能会成功的,这个想法感觉有一个巨大的瑕疵,但不知道為什麼它是work的,而且还曾经一度得到,state of the art的结果,deep learning就是這麼神奇,

## Speech

那在语音的部分,你也完全可以使用,self-supervised learning的概念



你完全可以试著训练,语音版的BERT

那怎麼训练语音版的BERT呢,你就看看文字版的BERT,是怎麼训练的,譬如说做填空题,语音也可以做填空题,就把一段声音讯号盖起来,叫机器去猜盖起来的部分是什麼嘛,语音也可以预测接下来会出现的内容,讲GPT就是预测,接下来要出现的token嘛,那语音你也可以叫它预测,叫模型预测接下来会出现的声音去套,所以你也可以做语音版的GPT,不管是语音版的BERT,语音版的GPT,其实都已经有很多相关的研究成果了

## Speech GLUE - SUPERB

不过其实在语音上,相较于文字处理的领域,还是有一些比较缺乏的东西,那我认为现在很缺乏的一个东西,就是像GLUE这样子的benchmark corpus

在自然语言处理的领域,在文字上有GLUE这个corpus,我们在这门课的刚开头,这个投影片的刚开头,就告诉你说有一个,这个基准的资料库叫做GLUE,它里面有九个NLP的任务,今天你要知道BERT做得好不好,就让它去跑那九个任务在去平均,那代表这个self-supervised learning,模型的好坏

但在语音上 到目前为止,还没有类似的基准的资料库,所以我们实验室就跟其他的研究团队,共同开发了一个语音版的GLUE

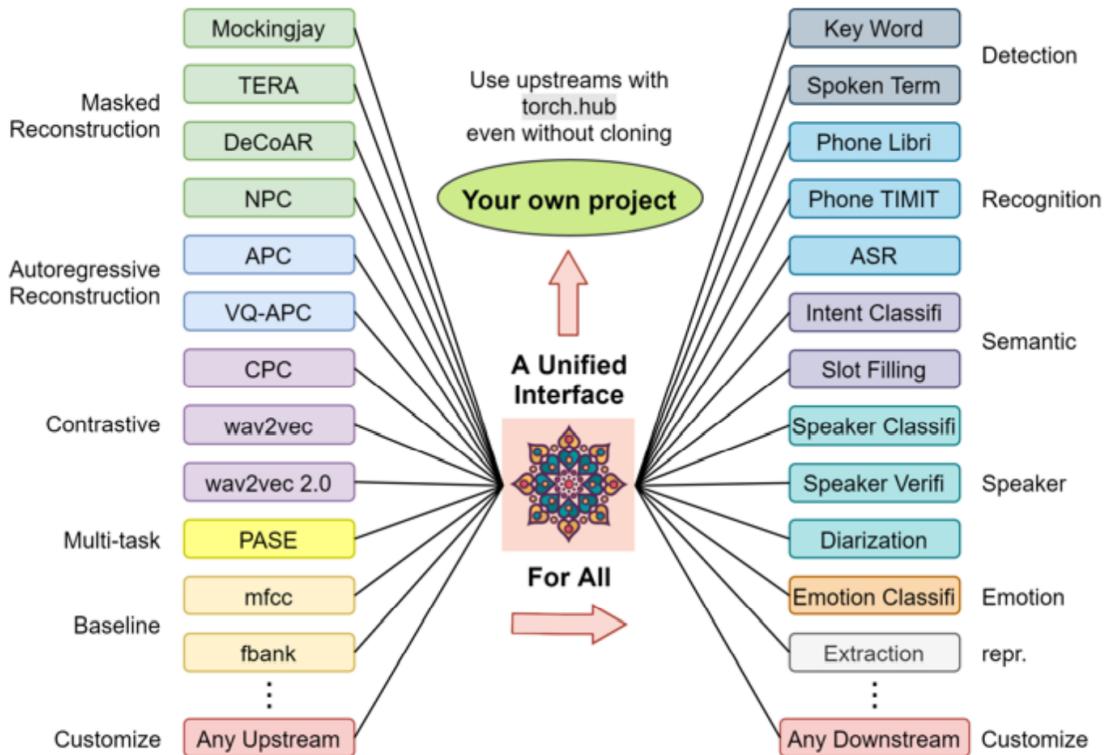
- **Speech processing Universal PERformance Benchmark**
  - Will be available soon
- **Downstream:** Benchmark with 10+ tasks
  - The models need to know how to process content, speaker, emotion, and even semantics.
- **Toolkit:** A flexible and modularized framework for self-supervised speech models.
  - <https://github.com/s3prl/s3prl>

我们叫做SUPERB,它是Speech processing Universal,PERformance Benchmark的缩写,你知道今天你做什麼模型,都一定要硬凑梗才行啦,所以这边也是要硬凑一个梗,把它叫做SUPERB

那其实我们已经准备了差不多了,其实网站都已经做好了,只等其他团队的人看过以后,就可以上线了,所以现在虽然还没有上线,但是再过一阵子,你应该就可以找得到相关的连结

在这个基准语料库裡面,包含了十个不同的任务,那语音其实有非常多不同的面向,很多人讲到语音相关的技术,都只知道语音辨识把声音转成文字,但这并不是语音技术的全貌,语音其实包含了非常丰富的资讯,它除了有内容的资讯,就是你说了什麼,还有其他的资讯,举例来说这句话是谁说的,举例这个人说这句话的时候,他的语气是什麼样,还有这句话背后,它到底有什麼样的语意,所以我们准备了十个不同的任务,这个任务包含了语音不同的面向,包括去检测一个模型,它能够识别内容的能力,识别谁在说话的能力,识别他是怎麼说的能力,甚至是识别这句话背后语意的能力,从全方位来检测一个, self-supervised learning的模型,它在理解人类语言上的能力

而且我们还有一个Toolkit,这个Toolkit裡面就包含了,各式各样的, self-supervised learning的模型



<https://github.com/andi611/Self-Supervised-Speech-Pretraining-and-Representation-Learning>

还有这些, self-supervised learning的模型, 它可以做的, 各式各样语音的下游的任务, 然后把连结放在这边给大家参考

讲这些只是想告诉大家说, self-supervised learning的技术, 不是只能被用在文字上, 在这个影像上 在语音上, 都仍然有非常大的空间可以使用, self-supervised learning的技术, 好那这个, self-supervised learning的部分呢, 这个BERT跟GPT我们就讲到这边,