

## AI 特許紹介(33)

AI 特許を学ぶ！究める！

～REALM(検索拡張言語モデル)の事前トレーニング特許～

2021 年 10 月 8 日

河野特許事務所

所長弁理士 河野英仁

「AI 特許紹介」シリーズは、注目すべき AI 特許のポイントを紹介します。熾烈な競争となっている第 4 次産業革命下では AI 技術がキーとなり、この AI 技術・ソリューションを特許として適切に権利化しておくことが重要であることは言うまでもありません。

AI 技術は Google, Microsoft, Amazon を始めとした IT プラットフォーマ、研究機関及び大学から毎週のように新たな手法が提案されており、また AI 技術を活用した新たなソリューションも次々とリリースされています。

本稿では米国先進 IT 企業を中心に、これらの企業から出願された AI 特許に記載された AI テクノロジー・ソリューションのポイントをわかりやすく解説致します。

### 1.概要

特許権者 Google

出願日 2020 年 5 月 20 日

登録日 2021 年 5 月 21 日

公開番号 US11003865

発明の名称 検索拡張言語モデルの事前トレーニングとファインチューニング

865 特許は、テキストの知識コーパス Z (Wikipedia 全体など) から知識を取得するニューラル知識検索器を使用して、言語モデルを教師無しで事前トレーニングするアイデアに関する。

### 2.特許内容の説明

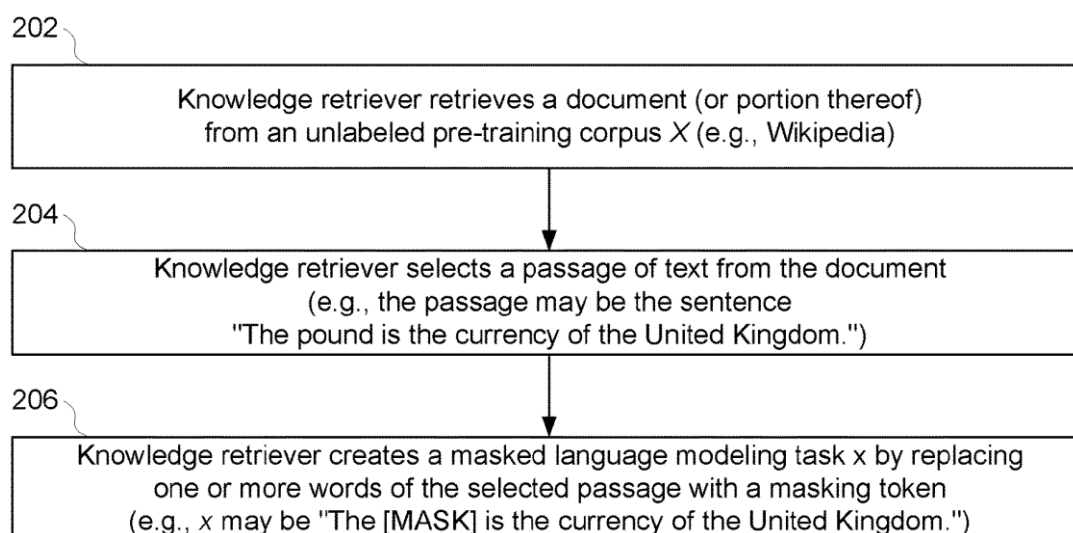
質問応答などの自然言語処理 (NLP) タスクは、世界の知識に基づいて事前にトレーニングされた言語モデルに依存している。言語モデルの事前トレーニングの最近の進歩により、トランスフォーマー双方向エンコーダー表現 (BERT: Bidirectional Encoder Representations from Transformers) 及びテキストからテキストへの転送トランスフ

オーマ（「T5」Text-to-Text Transfer Transformer）等の言語モデルは、トレーニング対象の大規模なテキストコーパスから取得した大量の世界知識を取り込むことができる。

しかしながら、言語モデルがより多くの知識を蓄積するにつれ、その知識をニューラルネットワークのパラメータに格納すれば、ネットワークのサイズが大幅に増加し、システム動作に悪影響を与えることとなる。

本発明は REALM（Retrieval-augmented language model）に対する事前トレーニングを教師無しで行う。学習信号としてマスクされた言語モデリングを使用し、数百万のドキュメントを考慮する検索ステップをバックプロパゲーションする。

図 2 はマスクされた言語モデルを生成するための知識検索器(Knowledge Retriever)による処理を示すフローチャートである。



ステップ 202 において、知識検索器は、ラベル付けされていない事前訓練コーパス X から文書を検索する。たとえば、事前トレーニングコーパスは、ウィキペディアなどのオンライン百科事典であり、取得されるドキュメントは、特定のエントリ、選択されたセクション、またはページのセクション（タイトル、本文、表など）、単一の段落または文などの完全な HTML ページである。

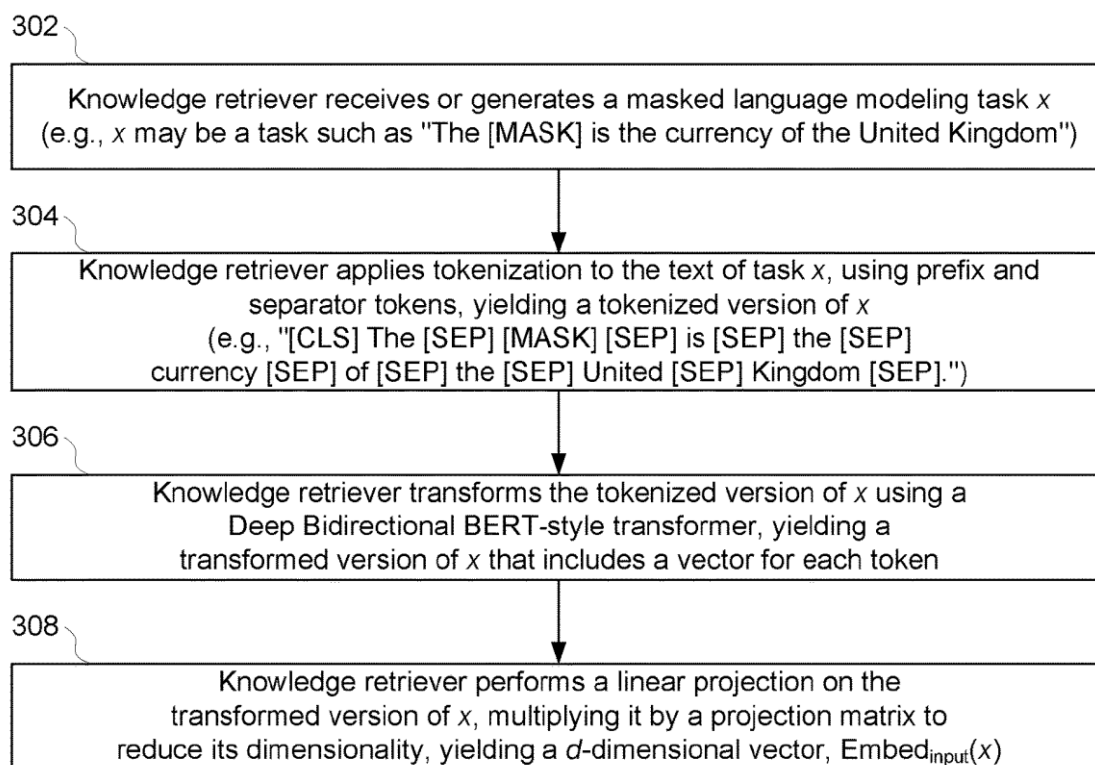
ステップ 204 において、知識検索器は、マスクされる文書からテキストの一節を選択する。たとえば、知識検索器は、“The pound is the currency of the United Kingdom.” 等、ドキュメントから 1 つの文を選択する。

最後に、ステップ 206 において、知識検索器は、選択された一節の 1 つまたは複数の単語をマスキングトークン（例えば、[MASK]）で置き換えることによって、マスクされた言語モデリングタスク  $x$  を作成する。

例えば、知識検索器は、マスクされた言語モデリングタスク  $x$  が、“The [MASK] is the currency of the United Kingdom.”となるように、選択されたパッセージ内の”pound”をマスクする。

知識検索器は、タスク  $x$  と、ラベルのないナレッジコーパス  $Z$  のドキュメントの両方を処理して、欠落しているトークンを予測する際に言語モデルで使用される  $k$  個の潜在的に関連するドキュメントのセットを識別する。

図 3 は、学習された埋め込み関数を使用して、マスクされたモデリングタスク  $x$  をベクトル化する処理を示すフローチャートである。



知識検索器がマスクされたモデリングタスク  $x$  を受信または生成した後（ステップ 302）、知識検索器は、タスク  $x$  のテキストをトークン化して、ステップ 304 で  $x$  のトークン化されたバージョンを作成する。

$x$  は、プレフィックストークン（[CLS]など）で始まり、 $x$  の各単語の後にセパレータ

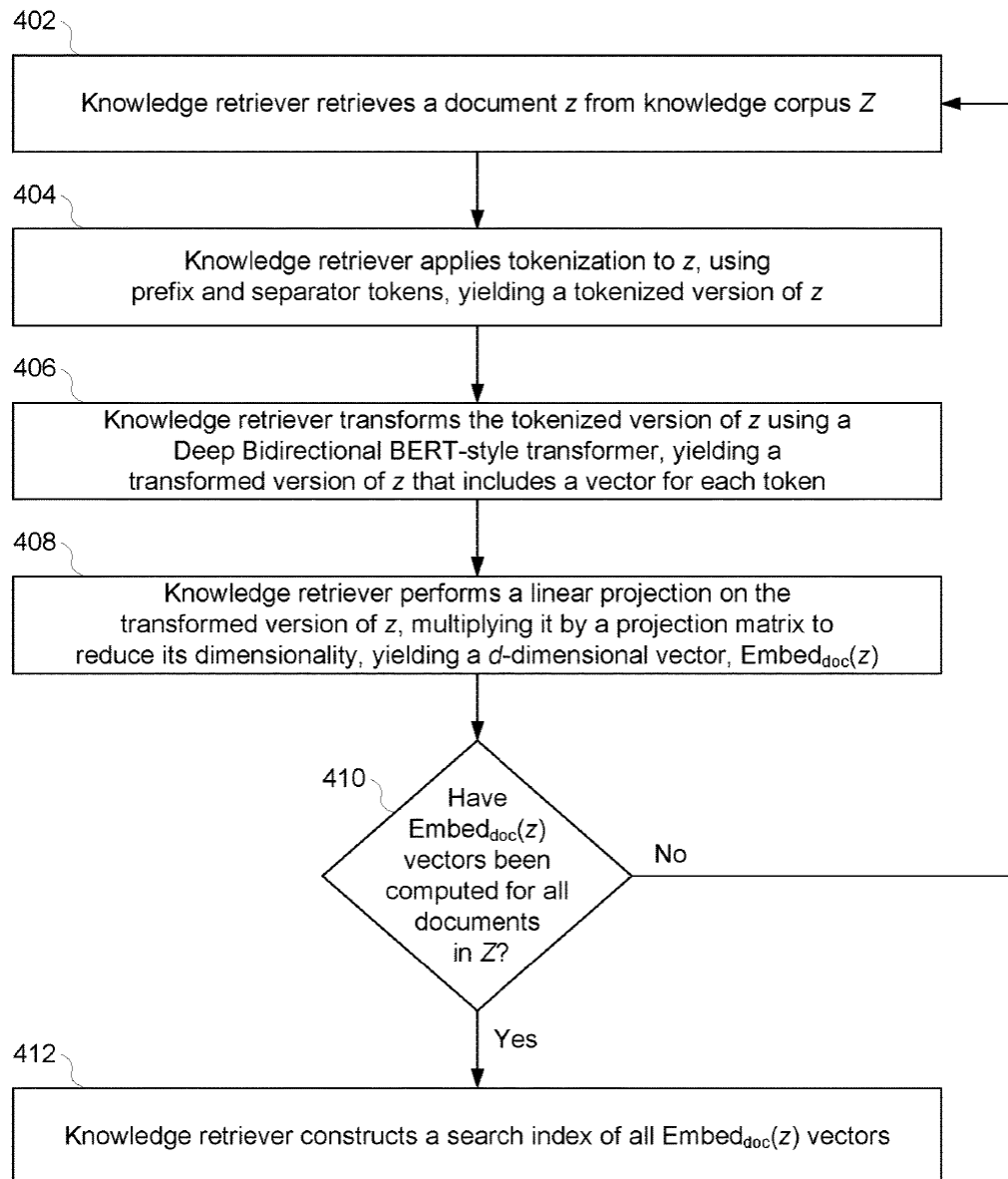
トークン ([SEP]など) が続くようにトークン化される。

$x$  が一節「The [MASK] is the currency of the United Kingdom」である場合、 $x$  のトークン化されたバージョンは、“[CLS] The [SEP] [MASK] [SEP] is [SEP] the [SEP] currency [SEP] of [SEP] the [SEP] United [SEP] Kingdom [SEP].”となる。

ステップ 306 において、知識検索器は、ディープ双方向 BERT スタイルのトランスフォーマーを使用して、トークン化されたバージョンの  $x$  を変換する。これにより、各トークンのベクトルを含む  $x$  の変換バージョンが生成される。

最後に、ステップ 308 において、知識検索器は、変換されたバージョンの  $x$  に対して線形射影を実行する。図 3 の例において、これは、 $x$  の変換されたバージョンに射影行列を乗算して、その次元を  $d$  次元ベクトル Embedinput ( $x$ ) に減らすことによって実現される。

図 4 は、学習された埋め込み関数を使用して、知識コーパス  $Z$  内の各文書  $z$  をベクトル化する処理を示すフローチャートである。



知識検索器は、ステップ 402 で、知識コーパス  $Z$  から文書  $z$  を検索することによって開始する。ナレッジコーパス  $Z$  は、トークンを予測する際に言語モデルによって使用される情報ソースである。たとえば、ナレッジコーパス  $Z$  は、ウィキペディアやその他の Web サイトなどのラベルのないコーパスである。

ステップ 404, 406, 408 において、知識検索器は、図 3 のステップ 304, 306, 308 で実行されたのと同じプロセスをトークン化、変換、および線形投影のプロセスを実行する。これらのステップは、ドキュメント  $z$  内の単語のすべてまたはサブセットに対して実行される。

ステップ 404 において、知識検索器は、タスク  $z$  のテキストをトークン化して、トークン化されたバージョンの  $z$  を作成する。この場合も、 $z$  はワードピーストークン化を使用してトークン化されるため、プレフィックストークン ([CLS] など) で始まり、 $z$  の各単語（またはそのセグメント化されたワードピース）の後にセパレータトークン ([SEP] など) が続く。

ステップ 406 において、知識検索器は、ディープ双方向 BERT スタイルのトランスフォーマーを使用して、トークン化されたバージョンの  $z$  を変換し、各トークンのベクトルを含む変換されたバージョンの  $z$  を生成する。

ステップ 408 において、知識検索器は、それを射影行列で乗算することによって、変換されたバージョンの  $z$  に対して線形射影を実行して、その次元を  $d$  次元ベクトル、Embeddoc( $z$ ) に低減する。

ステップ 410 において、知識検索器は、知識コーパス  $Z$  内のすべての文書について Embeddoc( $z$ ) ベクトルを計算したかどうかを決定する。そうでない場合、知識検索器は、ステップ 402 に戻り、そこで、知識コーパス  $Z$  から次の文書を選択し、ステップ 404, 406, 408 に関して説明したように、新しい文書  $z$  の処理に進む。

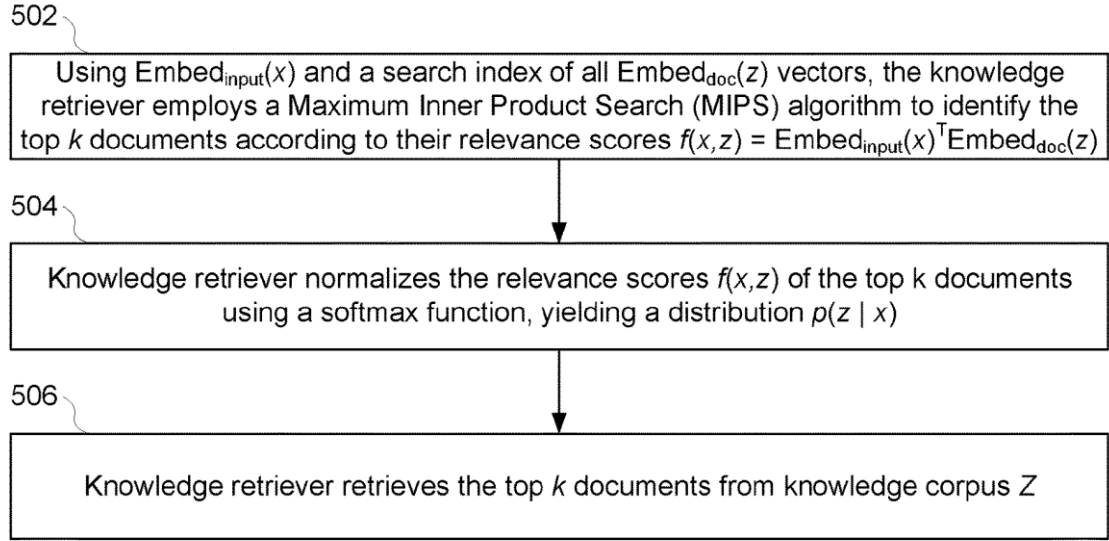
このプロセスは、ステップ 410 で、知識検索器が、知識コーパス  $Z$  内のすべての文書について Embeddoc( $z$ ) ベクトルが計算されたと決定するまで繰り返される。次に、ステップ 412 で、知識検索器は、計算されたすべての Embeddoc( $z$ ) ベクトルの検索インデックスを構築する。

知識検索器は、図 3 で作成されたベクトル Embedinput ( $x$ ) の転置と図 4 のステップ 408 で作成された特定の Embeddoc ( $z$ ) ベクトルの内積を取ることで、特定のドキュメント  $z$  の関連性スコアを計算する。所与のマスクされたモデリングタスク  $x$  について、所与の文書  $z$  の関連性スコア  $f(x, z)$  は、以下の式 1 に従って定義される。

$$f(x, z) = \text{Embed}_{input}(x)^T \text{Embed}_{doc}(z) \quad (1)$$

式 1 は、図 3 に従って作成された Embedinput ( $x$ ) ベクトルおよびすべての Embeddoc ( $z$ ) ベクトルの検索インデックスと組み合わせて使用して、特定のマスクされたモデリングタスク  $x$  の上位  $k$  個の関連ドキュメントを識別する。

図 5 は、例示的なプロセス 500 のフローチャートである。



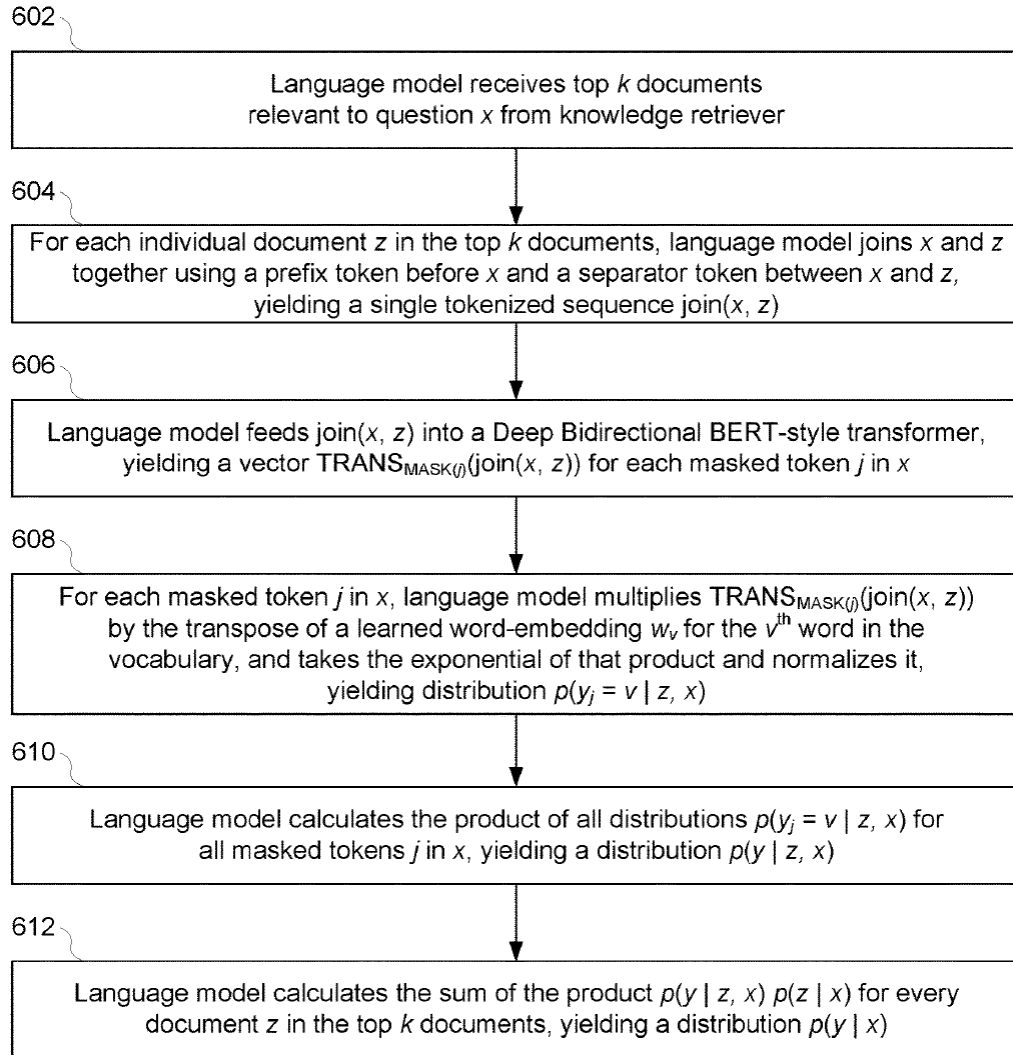
知識検索器は、 $\text{Embed}_{\text{input}}(\mathbf{x})$ および全ての  $\text{Embed}_{\text{doc}}(\mathbf{z})$ ベクトルの検索インデックスを使用して上位  $k$  個の文書を識別および検索する。ステップ 502 において、知識検索器は、図 3 に従って作成された  $\text{Embed}_{\text{input}}(\mathbf{x})$ ベクトル、図 4 のステップ 412 に従って作成されたすべての  $\text{Embed}_{\text{doc}}(\mathbf{z})$ ベクトルの検索インデックスを使用する最大内積検索 (MIPS) アルゴリズムを用い、個々の関連性スコア  $f(\mathbf{x}, \mathbf{z})$  に従って、上位  $k$  個の関連ドキュメントを識別する。これらの個々の関連性スコアは、上記の式 1 に従って計算される。

ステップ 504 において、知識検索器は、ソフトマックス関数を使用して上位  $k$  個の文書の関連性スコア  $f(\mathbf{x}, \mathbf{z})$  を正規化して、分布  $p(\mathbf{z}|\mathbf{x})$  を生成する。より具体的には、上位  $k$  個のドキュメントのセット内の各ドキュメント  $\mathbf{z}$  について、分布  $p(\mathbf{z}|\mathbf{x})$  は、以下の式 2 で示されているように、そのドキュメントの関連性スコアの指数を上位  $k$  個のドキュメントのそれぞれの関連性スコアの合計で割ったものになる。

$$p(\mathbf{z}|\mathbf{x}) = \frac{\exp f(\mathbf{x}, \mathbf{z})}{\sum_{\mathbf{z}'} \exp(\mathbf{x}, \mathbf{z}')} \quad (2)$$

最後に、ステップ 506 において、知識検索器は、言語モデルによって使用するために上位  $k$  個の文書のそれぞれを検索する。言語モデルが上位  $k$  個のドキュメントを受信すると、言語モデルと知識検索器は次の手順を実行して、 $k$  個の取得されたドキュメントのセットに基づいて特定の応答  $\mathbf{y}$  を生成する可能性を識別する。

図 6 は、事前トレーニング中の分布  $p(\mathbf{y}|\mathbf{x})$  の生成処理を示すフローチャートである。



ステップ 602 において、言語モデルは、マスクされたモデリングタスク  $x$  に関連する上位  $k$  個の文書を知識検索器から受け取る。ステップ 604 において、上位  $k$  個のドキュメント内の個々のドキュメント  $z$  ごとに、言語モデルは、 $x$  の前にプレフィックストークン ([CLS] など) と  $x$  と  $z$  の間にセパレータトークン ([SEP] など) を使用して、 $x$  と  $z$  を 1 つのシーケンスに結合する。これにより、ドキュメント  $z$  ごとに単一のトークン化されたシーケンス  $\text{join}(x, z)$  が作成される。

例えば、 $x$  が文章 “The [MASK] is the currency of the United Kingdom,” であり、 $z$  が文章 “Pound sterling, known in some contexts simply as the pound or sterling, is the official currency of the United Kingdom,” (英ポンドは、一部の文脈では単に英ポンドまたは英ポンドとして知られ、英国の公式通貨です。) である場合、トークン化されたシーケンス  $\text{join}(x, z)$  は、: “[CLS] The [MASK] is the currency of the United

Kingdom [SEP] Pound sterling, known in some contexts simply as the pound or sterling, is the official currency of the United Kingdom.” となる。

ステップ 606 において、言語モデルは、各  $\text{join}(\mathbf{x}, \mathbf{z})$  シーケンスをトランスフォーマーにフィードして、シーケンス内の各マスクされたトークンのベクトルを作成する。これにより、 $\mathbf{x}$  内のマスクされたトークン  $j$  ごとにベクトル  $\text{TRANS}_{\text{MASK}(j)}(\text{join}(\mathbf{x}, \mathbf{z}))$  が生成される。

ステップ 608 において、言語モデルは、そのマスクされたトークンのために学習された単語埋め込みの転置によって、 $\text{TRANS}_{\text{MASK}(j)}(\text{join}(\mathbf{x}, \mathbf{z}))$  ベクトルを乗算する。語彙の  $v$  番目の単語の単語埋め込みは、語彙が  $v=\{1, \dots, V\}$  エントリである  $\mathbf{w}_v$  として表される。

埋め込み  $\mathbf{w}_v$  は、言語モデルの語彙に含まれるすべての既知の単語または単語のベクトルを含む、学習された単語埋め込みマトリックスからその単語または単語を表すベクトルになる。 $\mathbf{w}_v$  の転置に  $\text{TRANS}_{\text{MASK}(j)}(\text{join}(\mathbf{x}, \mathbf{z}))$  を掛けると、言語モデルの語彙と同じ次元のベクトルが得られる。

次に、言語モデルは、結果のベクトルの指数を取得し、1 から  $V$  までのすべてのエンタリ  $v$  のベクトルの要素の合計で除算することによって正規化し、下記式(3)に示すように、 $\mathbf{z}$  と  $\mathbf{x}$  を観察して、 $j$  番目のマスクされたトークンの真の値が語彙の  $v$  番目の単語に等しい確率を表す分布  $p(y_j=v | \mathbf{z}, \mathbf{x})$  を生成する。

$$p(y_j = v | \mathbf{z}, \mathbf{x}) = \frac{\exp(\mathbf{w}_v^T \text{TRANS}_{\text{MASK}(j)}(\text{join}(\mathbf{x}, \mathbf{z})))}{\sum_{v'=1}^V \exp(\mathbf{w}_{v'}^T \text{TRANS}_{\text{MASK}(j)}(\text{join}(\mathbf{x}, \mathbf{z})))} \quad (3)$$

ステップ 610 において、言語モデルは、 $\mathbf{x}$  の各マスクされたトークン  $j$  の分布  $p(y_j=v | \mathbf{z}, \mathbf{x})$  を計算するために、 $\mathbf{x}$  の各マスクされたトークンについて上記の式 3 の計算を繰り返す。言語モデルは、与えられたドキュメント  $\mathbf{z}$  に基づき、これらの各分布  $p(y_j | \mathbf{z}, \mathbf{x})$  の積を取り、 $\mathbf{x}$  内のすべてのマスクされたトークンの各可能な出力に関する予測を表す最終的な分布  $p(\mathbf{y} | \mathbf{z}, \mathbf{x})$  を生成する。 $J$  個のマスクされたトークンを持つシーケンス  $\mathbf{x}$  の場合、最終的な分布  $p(\mathbf{y} | \mathbf{z}, \mathbf{x})$  は、以下の式 4 に従って定義される。

$$p(\mathbf{y} | \mathbf{z}, \mathbf{x}) = \prod_{j=1}^J p(y_j = v | \mathbf{z}, \mathbf{x}) \quad (4)$$

最後に、ステップ 612 において、言語モデルは、図 6 のステップ 610 および図 5 の

ステップ 504 で作成された確率分布を使用して、第 3 の確率分布  $p(y|x)$  を生成する。確率分布  $p(y|x)$  は、上位  $k$  個のドキュメント内のすべてのドキュメント  $z$  に基づいて、 $x$  内のすべてのマスクされたトークンの各可能な出力の全体的な可能性を表す。

この分布は、 $z$  を潜在変数として扱い、すべてのドキュメント  $z$  (上位  $k$  個のドキュメント) をマージナル化することによって生成される。したがって、 $p(y|x)$  は、以下の式 5 に示すように、上位  $k$  個のドキュメント内のすべてのドキュメント  $z$  の積  $p(y|z, x) p(z|x)$  の合計をとることによって計算される。

$$p(y|x) = \sum_{z \in Z} p(y|z, x) p(z|x) \quad (5)$$

言語モデルが上位  $k$  個のドキュメント内のすべてのドキュメント  $z$  の  $p(y|x)$  を決定すると、知識検索器はその分布を使用して  $\text{Embed}_{\text{input}}$  と  $\text{Embed}_{\text{doc}}$  のパラメータを更新できる。その点で、知識検索器は  $\text{Embed}_{\text{input}}$  と  $\text{Embed}_{\text{doc}}$  のパラメータを更新して、対数尤度  $\log p(y|x)$  が正しい出力  $y$  に対して最大化されるようにする。

知識検索器は、確率的勾配降下法を使用する。対数尤度の勾配  $\nabla \log p(y|x)$  は、特定のドキュメント  $z$  が  $Z$  の平均ドキュメントよりも情報量が多いか少ないかに基づいて、各ドキュメント  $z$  の個々の関連性スコア  $f(x, z)$  が増加または減少するにつれて、繰り返し再計算される。知識検索器は、以下の式 6 に従って定義される乗数  $r(z)$  によって、各文書  $z$  の関連性スコア  $f(x, z)$  を増加または減少させることができる。

$$r(z) = \left[ \frac{p(y|z, x)}{p(y|x)} - 1 \right] p(z|x) \quad (6)$$

$f(x, z)$  は  $\text{Embed}_{\text{input}}(x)$  と  $\text{Embed}_{\text{doc}}(z)$  の転置の内積であるため、知識検索器は、 $\text{Embed}_{\text{input}}$  および  $\text{Embed}_{\text{doc}}$  埋め込み関数のパラメータを変更することにより、これらの増加または減少を  $f(x, z)$  に実装する。知識検索器は、 $\log p(y|x)$  が最大化されるまで、 $\text{Embed}_{\text{input}}$  および  $\text{Embed}_{\text{doc}}$  埋め込み関数のパラメータを改訂し、 $\nabla \log p(y|x)$  を再計算し続ける。

### 3. クレーム

865 特許のクレーム 1 は以下の通りである。

#### 1. 言語モデルをトレーニングするコンピュータ実装の方法において、

処理システムの 1 つまたは複数のプロセッサを使用して、第 1 の文書からのテキストを使用してマスクされた言語モデリングタスクを生成し、

1 つまたは複数のプロセッサを使用して、マスクされた言語モデリングタスクに第 1 の学習された埋め込み関数を適用することにより、入力ベクトルを生成し、

1 つまたは複数のプロセッサを使用して、知識コーパスの各文書に第 2 の学習された埋め込み関数を適用することにより、知識コーパスの各文書の文書ベクトルを生成し、前記知識コーパスは、第 1 の複数の文書を含み、

1 つまたは複数のプロセッサを使用して、前記入力ベクトルおよび所与の文書の前記文書ベクトルに基づいて、知識コーパスの所与の各文書の関連性スコアを生成し、

1 つまたは複数のプロセッサを使用して、第 2 の複数の文書内の各文書の関連性スコアに基づいて第 1 の分布を生成し、第 2 の複数の文書は知識コーパスからのものであり、

1 つまたは複数のプロセッサを使用して、マスクされた言語モデリングタスクおよび第 2 の複数の文書内の各文書のテキストに基づいて第 2 の分布を生成し、

1 つまたは複数のプロセッサを使用して、第 1 の分布および第 2 の分布に基づいて第 3 の分布を生成し、

1 つまたは複数のプロセッサを使用して、更新された第 1 の分布および更新された第 3 の分布を生成するために、少なくとも第 1 の学習された埋め込み関数または第 2 の学習された埋め込み関数の 1 つまたは複数のパラメータを変更する。

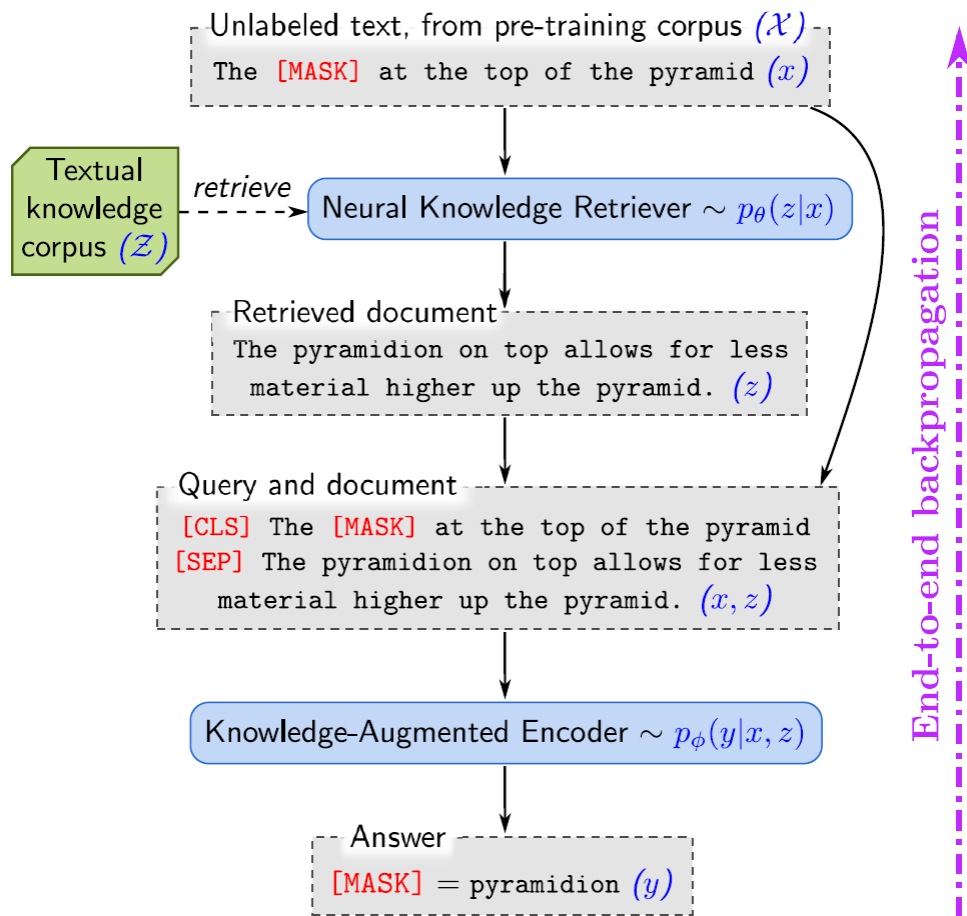
#### 4. 本特許に関する論文

本特許に関連する論文<sup>1</sup>「REALM: Retrieval-Augmented Language Model Pre-Training」が Kelvin Guu 氏らにより発表されている。

下記図 1 は教師無しの事前トレーニング処理を示すフローチャートである。

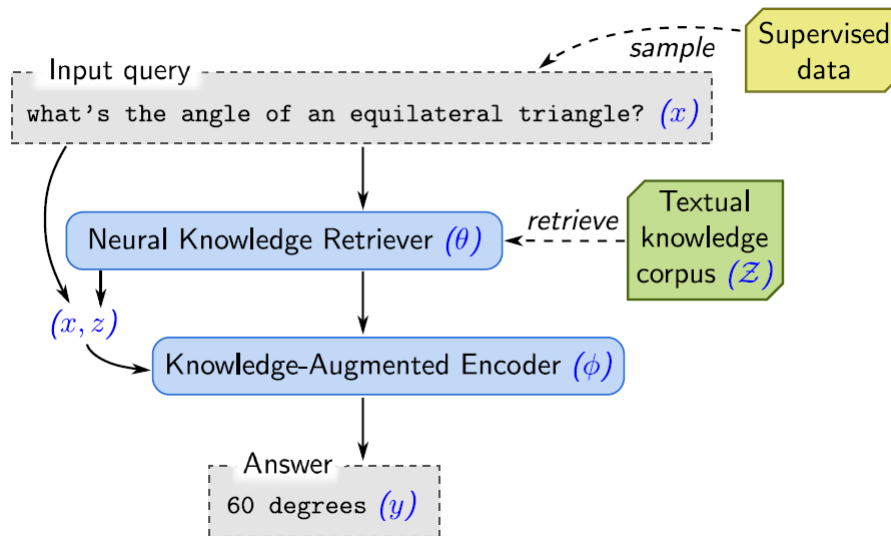
---

<sup>1</sup> Kelvin Guu , Kenton Lee, Zora Tung, Panupong Pasupat, Ming-Wei Chang  
“REALM: Retrieval-Augmented Language Model Pre-Training”  
arXiv:2002.08909v1 [cs.CL] 10 Feb 2020



REALM は、テキストの知識コーパス  $Z$  (Wikipedia 全体など) から知識を取得するニューラル知識検索器を使用して、言語モデルを事前トレーニングする。処理内容は特許の内容と同様である。

下記図 2 は教師ありファインチューニング処理を示すフローチャートである。



事前トレーニングの後に、Q&A等の各タスクに応じて教師あり学習によりファインチューニングされる。

下記テーブルは、Open-QA ベンチマークでのテスト結果を示す。

| Name                                                         | Architectures              | Pre-training   | NQ<br>(79k/4k) | WQ<br>(3k/2k) | CT<br>(1k/1k) | # params |
|--------------------------------------------------------------|----------------------------|----------------|----------------|---------------|---------------|----------|
| BERT-Baseline (Lee et al., 2019)                             | Sparse Retr.+Transformer   | BERT           | 26.5           | 17.7          | 21.3          | 110m     |
| T5 (base) (Roberts et al., 2020)                             | Transformer Seq2Seq        | T5 (Multitask) | 27.0           | 29.1          | -             | 223m     |
| T5 (large) (Roberts et al., 2020)                            | Transformer Seq2Seq        | T5 (Multitask) | 29.8           | 32.2          | -             | 738m     |
| T5 (11b) (Roberts et al., 2020)                              | Transformer Seq2Seq        | T5 (Multitask) | 34.5           | 37.4          | -             | 11318m   |
| DrQA (Chen et al., 2017)                                     | Sparse Retr.+DocReader     | N/A            | -              | 20.7          | 25.7          | 34m      |
| HardEM (Min et al., 2019a)                                   | Sparse Retr.+Transformer   | BERT           | 28.1           | -             | -             | 110m     |
| GraphRetriever (Min et al., 2019b)                           | GraphRetriever+Transformer | BERT           | 31.8           | 31.6          | -             | 110m     |
| PathRetriever (Asai et al., 2019)                            | PathRetriever+Transformer  | MLM            | 32.6           | -             | -             | 110m     |
| ORQA (Lee et al., 2019)                                      | Dense Retr.+Transformer    | ICT+BERT       | 33.3           | 36.4          | 30.1          | 330m     |
| Ours ( $\mathcal{X}$ = Wikipedia, $\mathcal{Z}$ = Wikipedia) | Dense Retr.+Transformer    | REALM          | 39.2           | 40.2          | <b>46.8</b>   | 330m     |
| Ours ( $\mathcal{X}$ = CC-News, $\mathcal{Z}$ = Wikipedia)   | Dense Retr.+Transformer    | REALM          | <b>40.4</b>    | <b>40.7</b>   | 42.9          | 330m     |

本論文におけるモデル REALM は、既存のすべてのシステムよりも優れていることが理解できる。

以上

著者紹介

河野英仁

河野特許事務所、所長弁理士。立命館大学情報システム学博士前期課程修了、米国フランクリンピアースローセンター知的財産権法修士修了、中国清華大学法学院知的財産夏季セミナー修了、MIT(マサチューセッツ工科大学)コンピュータ科学・AI 研究所 AI コース修了。

[AI 特許コンサルティング](#)、[医療 AI 特許コンサルティング](#)の他、米国・中国特許の権

利化・侵害訴訟を専門としている。著書に「世界のソフトウェア特許(共著)」、「FinTech 特許入門」、「[AI/IoT 特許入門 2.0](#)」、「[ブロックチェーン 3.0](#)(共著)」がある。