機械学習研究 21 (2020) 1-67 1/20提出; 6/20修正; 6/20発行

統一されたテキスト間変換装置で転移学習の限界に挑む

Colin Raffel* （ craffel@gmail.com
コリン・ラ
フェル
ノーム・シェイザー*。 NOAM@GOOGLE.COM
アダム・ロバーツ*。 ADAROB@GOOGLE.COM
キャサリン・リー*さん KATHERINELEE@GOOGLE.COM
シャラン・ナ SHARANNARANG@GOOGLE.COM
ラン
マイケル・マテナ mmatena@google.com
周燕起（しゅう・えんき YANQIZ@GOOGLE.COM
ウェイ・リ MWEILI@GOOGLE.COM
PETERJLIU@GOOGLE.COM
ー ピータ
ー・J・リ
ユー

グーグル、マウンテンビュー、カリ
フォルニア州94043、米国

編集者： イワン・ティトフ

アブストラクト

自然言語処理(NLP)分野では、データ量の多いタスクでモデルを事前学習した後に、下流のタスクで微調整を行う「転移学習」が強力な手法として注目されている。転移学習の有効性は、多様なアプローチ、方法論、実践を生んでいる。本稿では、すべてのテキストベースの言語問題をテキストからテキストに変換する統一的なフレームワークを導入することで、NLPのための転移学習技法の景観を探る。我々の体系的な研究では、数十の言語理解タスクについて、事前学習の目的、アーキテクチャ、ラベルなしデータセット、転送アプローチ、その他の要素を比較する。私たちの探求から得た知見と規模、そして新しい「Colossal Clean Crawled Corpus」を組み合わせることで、要約、質問応答、テキスト分類などをカバーする多くのベンチマークにおいて、最先端の結果を得ることができました。NLPのための転移学習に関する今後の研究を促進するため、データセット、事前学習済みモデル、コードを公開します[1]。

キーワード：転移学習、自然言語処理、マルチタスク学習、アテンションベースモデル、ディープラーニング

1. はじめに

自然言語処理（NLP）タスクを実行するための機械学習モデルのトレーニングには、しばしば、モデルが下流学習が可能な方法でテキストを処理できることが必要です。これは、モデルがテキストを「理解」できるようにするための汎用的な知識を開発することと大まかに考えることができる。この知識は、低レベルのものから（例えば、スペルや文法、単語、文法など）様々です。

---

*.平等な貢献。各著者の貢献度の説明は、付録Aに掲載されている。通信 から craffel@gmail.com.

1. https://github.com/google-research/text-to-text-transfer-transformer

©2020 Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu.

ライセンスCC-BY 4.0、https://creativecommons.org/licenses/by/4.0/ を参照してください。表示条件は http://jmlr.org/papers/v21/20-074.html に記載されています。

例えば、チューバは大きすぎてほとんどのバックパックに入らないなど）からハイレベルなものまで。現代の機械学習の実践では、この知識を提供することが明示的に行われることはほとんどなく、代わりに、補助的なタスクの一部として学習されることが多い。例えば、歴史的に一般的なアプローチは、単語ベクトル(Mikolov et al., 2013b,a; Pennington et al., 2014)を使用して、単語の同一性を連続的な表現にマッピングすることです。このベクトルは、例えば、共起語を連続空間の近くに配置するよう促す目的を通して学習されることが多い（Mikolov et al.、2013b）。

近年では、データ量の多いタスクでモデル全体を事前学習させることが一般的になってきています。理想的には、この事前学習によって、モデルが汎用的な能力と知識を身につけ、それを下流のタスクに転送することができる。コンピュータビジョンへの転移学習の応用（Oquab et al., 2014; Jia et al., 2014; Huh et al., 2016; Yosinski et al., 2014）において、事前学習は、一般的にImageNet（Russakovsky et al., 2015; Deng et al., 2009）などのラベル付き大規模データセットでの教師あり学習によって行われます。これに対して、NLPにおける転移学習の最新の技術では、ラベルのないデータに対する教師なし学習を用いた事前学習が行われることが多い。このアプローチは最近、最も一般的なNLPベンチマークの多くで最先端の結果を得るために使用されている（Devlin et al., 2018; Yang et al., 2019; Dong et al., 2019; Liu et al., 2019c; Lan et al., 2019）。その経験的な強さだけでなく、NLPのための教師なし事前学習が特に魅力的なのは、インターネットによってラベルのないテキストデータが大量に入手できるからです。例えば、Common Crawlプロジェクト[2]は毎月ウェブページから抽出した約20TBのテキストデータを生成しています。これは、顕著なスケーラビリティを示すことが示されているニューラルネットワークに自然に適合しています。つまり、より大きなデータセットでより大きなモデルをトレーニングするだけで、より優れた性能を達成できることがよくあります（Hestness et al., 2017; Shazeer et al., 2017; Jozefowicz et al., 2016; Mahajan et al., 2018; Radford et al., 2019; Shazeer et al.）

この相乗効果により、NLPのための転移学習方法論を開発する最近の研究は非常

に多く、事前学習目的の幅広い風景を生み出している（Howard and Ruder, 2018; Devlin et al, 2018; Yang et al、2019）、ラベルなしデータセット（Yang et al., 2019; Liu et al., 2019c; Zellers et al., 2019）、ベンチマーク（Wang et al., 2019b, 2018; Conneau and Kiela, 2018）、微調整方法（Howard and Ruder, 2018; Houlsby et al., 2019; Peters et al., 2019）などなど。この急成長中の分野における急速な進歩の速度と技術の多様性は、異なるアルゴリズムの比較、新しい貢献の効果の切り分け、および転移学習のための既存の手法の空間の理解を困難にする可能性があります。より厳密な理解の必要性に突き動かされ、我々は、異なるアプローチを体系的に研究し、この分野の現在の限界を押し広げることを可能にする、転移学習への統一的なアプローチを活用しています。

私たちの仕事の基礎となる基本的な考え方は、すべてのテキスト処理問題を「テキストからテキストへ」の問題、すなわちテキストを入力として受け取り、新しいテキストを出力として生成するものとして扱うことです。このアプローチは、すべてのテキスト問題を質問応答（McCannら、2018）、言語モデリング（Radfordら、2019）、またはスパン抽出Keskarら（2019b）タスクとしてキャストするなど、これまでのNLPタスクの統一フレームワークに触発されています。重要なのは、text-to-textフレームワークにより、我々が考えるすべてのタスクに同じモデル、目的、トレーニング手順、デコードプロセスを直接適用することができることである。この柔軟性を活かして、質問応答、文書、音声、画像など、英語ベースのさまざまなNLP問題で性能を評価する。

2. http://commoncrawl.org

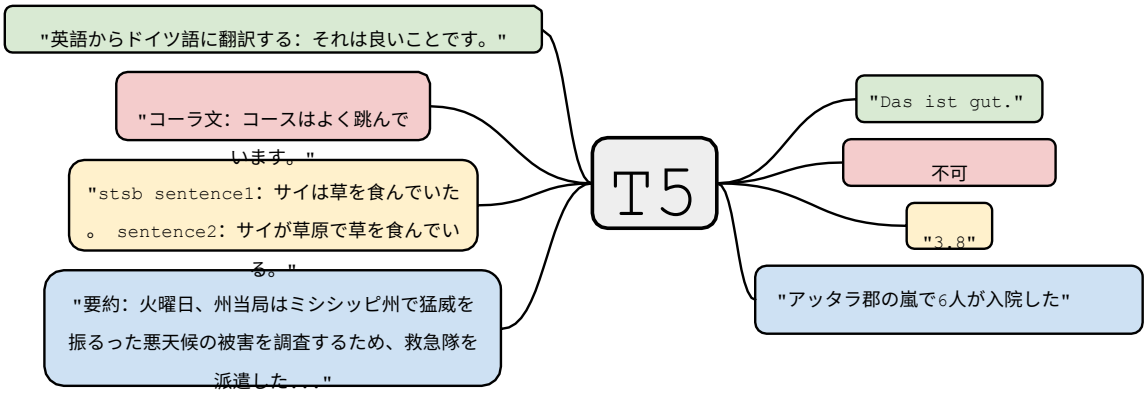

図1：Text-to-Textフレームワークの構成図。翻訳、質問応答、分類など、私たちが考えるすべてのタスクは、入力としてモデルのテキストを与え、それを訓練してターゲットテキストを生成するというものである。これにより、多様なタスクで同じモデル、損失関数、ハイパーパラメータなどを使用することができます。また、この実証実験に含まれる手法の標準的なテストベッドを提供します。T5」とは、「Text-to-Text Transfer Transformer」と名付けた我々のモデルのことで、このモデルは「Text-to-Text Transfer Transformer」と呼ばれています。

要約、感情分類などである。この統一的なアプローチにより、異なる転移学習の目的、ラベルのないデータセットなどの有効性を比較することができ、また、これまで考えられてきた以上にモデルやデータセットをスケールアップすることにより、NLPにおける転移学習の限界を探ることができます。

私たちの目標は、新しい手法を提案することではなく、この分野の立ち位置について包括的な視点を提供することであることを強調します。そのため、我々の研究は、主に既存の手法の調査、探索、実証的な比較から構成されています。また、我々の体系的な研究（最大110億のパラメータを持つモデルのトレーニング）から得られた知見を拡大し、我々が検討する多くのタスクで最先端の結果を得ることで

、現在のアプローチの限界を探ります。この規模の実験を行うために、ウェブからかき集めた数百ギガバイトのきれいな英文テキストからなるデータセット「Colossal Clean Crawled Corpus」（C4）を導入しました。データ不足の環境において、訓練済みのモデルを活用できることが転移学習の主な有用性であると認識し、コード、データセット、訓練済みモデルを公開しています[1]。

本論文の残りの部分は以下のように構成されている：次のセクションでは、我々の基本モデルとその実装、あらゆるテキスト処理問題をテキストからテキストへのタスクとして定式化するための手順、そして我々が考慮するタスク群について述べる。セクション3では、NLPのための転移学習の分野を探求する大規模な実験セットを提示する。セクションの最後（セクション3.7）では、我々の体系的な研究からの洞察を組み合わせて、多種多様なベンチマークで最先端の結果を得る。最後に、結果のまとめを行い、セクション4で将来に向けた展望を述べて締めくくられる。

2. セットアップ

大規模な実証研究の結果を紹介する前に、Transformerモデルのアーキテクチャや評価対象とした下流タスクなど、我々の結果を理解するために必要な背景トピックを確認します。また、あらゆる問題をテキストからテキストへのタスクとして扱う我々のアプローチを紹介し、ラベルのないテキストデータのソースとして我々が作成したCommon Crawlベースのデータセットである「Colossal Clean Crawled Corpus」（C4）について説明する。私たちのモデルとフレームワークを「Text-to-Text Transfer Transformer」(T5)と呼んでいます。

2.1. モデル

NLPのための転移学習に関する初期の結果は、リカレントニューラルネットワークを活用していましたが（Peters et al., 2018; Howard and Ruder, 2018）、最近では「Transformer」アーキテクチャに基づくモデル（Vaswani et al., 2017）を用いることが一般的になってきています。Transformerは当初、機械翻訳に有効であることが示されましたが、その後、さまざまなNLPの場面で使用されています（Radford et al., 2018; Devlin et al., 2018; McCann et al., 2018; Yu et al.、2018）。そのユビキタス性が高まっているため、私たちが研究するモデルはすべてTransformerアーキテクチャに基づいています。後述する詳細とセクション3.2で探索する変種を除けば、我々は元々提案されたこのアーキテクチャから大きく逸脱することはない。このモデルの包括的な定義を提供する代わりに、我々は、より詳細な導入のために、関心のある読者に元の論文（Vaswani et al.、2017）またはフォローアップチュートリアル[3,4]を参照する。

Transformerの主要なビルディングブロックは、自己注意です（Cheng et al.、2016）。自己注意は注意の変種であり（Graves, 2013; Bahdanau et al., 2015）、各要素をシーケンスの残りの要素の加重平均で置き換えることによってシーケンスを処理する。オリジナルのTransformerはエンコーダ-デコーダアーキテクチャで構成されており、シーケンス間（Sutskever et al., 2014; Kalchbrenner et al., 2014）タスクを対象としていた。また、最近では、単一のTransformer層スタックから

なるモデルを使用することが一般的になっており、言語モデリング（Radford et al., 2018; Al-Rfou et al., 2019）または分類とスパン予測タスク（Devlin et al., 2018; Yang et al., 2019）に適したアーキテクチャを作り出すためにさまざまな形態の自己アテンションを使用しました。セクション3.2では、これらのアーキテクチャの変種を経験的に探索する。

全体として、我々のエンコーダ・デコーダTransformerの実装は、その元々提案された形（Vaswani et al, 2017）に忠実に従う。まず、トークンの入力シーケンスがエンベッディングのシーケンスにマッピングされ、それがエンコーダーに渡される。エンコーダは「ブロック」のスタックで構成され、各ブロックは2つのサブコンポーネントで構成される：自己注意層の後に小さなフィードフォワードネットワークがある。各サブコンポーネントの入力には、レイヤー正規化（Ba et al.、2016）が適用される。我々は、活性度が再スケールされるだけで、加算バイアスが適用されない、層正規化の簡略版を使用する。レイヤー正規化の後、残差スキップ接続（He et al.、2016）が各サブコンポーネントの入力をその出力に加える。ドロップアウト（Srivastava et al., 2014）は、フィードフォワードネットワーク内、スキップ接続上、注目重み上、およびスタック全体の入力と出力で適用される。デコーダは、標準的なアテンションウェイトを含む以外は、エンコーダと同様の構造である。

3. http://nlp.seas.harvard.edu/2018/04/03/attention.html
4. http://jalammar.github.io/illustrated-transformer/

エンコーダの出力に注意を向ける各自己注意層の後に、メカニズムがあります。デコーダーの自己注意メカニズムは、自己回帰的または因果的な自己注意の形式も用いており、このモデルは過去の出力にのみ注意することができる。最後のデコーダブロックの出力は、ソフトマックス出力の密な層に供給され、その重みは入力埋め込み行列と共有される。Transformerのすべての注意メカニズムは、独立した「ヘッド」に分割され、その出力は、さらに処理される前に連結される。

自己アテンションは順序に依存しない（つまり集合に対する操作である）ので、Transformerに明示的な位置信号を提供することが一般的である。当初のTransformerは正弦波状の位置信号や学習済みの位置埋め込みを使用していましたが、最近では相対位置埋め込みを使用することが一般的になっています（Shaw et al., 2018; Huang et al., 2018a）。相対位置埋め込みは、各位置に固定された埋め込みを用いるのではなく、自己注意メカニズムにおいて比較される「キー」と「クエリ」のオフセットに応じて異なる学習済み埋め込みを生成する。我々は、位置埋め込みを簡略化し、各埋め込みは単にスカラーであり、注意の重みを計算するために使用される対応するロジットに追加される形式を用いている。また、効率化のため、位置埋め込みパラメータをモデルの全層で共有するが、ある層では各注意ヘッドが異なる学習済み位置埋め込みを使用する。一般に、一定の数の埋め込みが学習され、それぞれがキーとクエリのオフセットの可能性の範囲に対応する。この研究では、すべてのモデルで32の埋め込みを使用し、128のオフセットまで対数的にサイズが増加する範囲を設定し、それ以降はすべての相対位置を同じ埋め込みに割り当てている。ある層は128トークンを超える相対位置には鈍感であるが、後続の層は前の層からの局所情報を組み合わせることで、より大きなオフセットに対する感度を構築することができることに注意。要約すると、我々のモデルは、Layer Normバイアスを削除し、レイヤー正規化を残差パスの外に配置し、異なる位置埋め込みスキームを使用することを除いて、Vaswaniら（2017）が提案したオリジナルのTransformerとほぼ同等である。これらのアーキテクチャの変更は、我々が転移学習の実証的調査で考慮する実験的要因と直交しているため、その影響の切除は今後の作業に委ねます。

本研究の一環として、これらのモデルのスケーラビリティ、すなわち、より多く

のパラメータやレイヤーを持つようにした場合に性能がどのように変化するかを実験しています。大規模なモデルのトレーニングは、1台のマシンに収まらず、膨大な計算を必要とするため、非自明な場合があります。TPUポッドとは、1,024個のTPU v3チップを搭載し、CPUホストマシンと高速2Dメッシュインターコネクトで接続されたマルチラックMLスーパーコンピュータである[5]。モデル並列とデータ並列の両方を簡単に実装できるように、Mesh TensorFlowライブラリ（Shazeerら、2018）を活用します（Krizhevsky、2014）。

2.2. 巨大なクリーンクロールドコーパス

NLPのための転移学習に関する先行研究の多くは、教師なし学習のために大規模なラベルなしデータセットを利用している。本論文では、このラベルなしデータの品質、特性、サイズの効果を測定することに関心がある。我々のニーズを満たすデータセットを生成するために、ウェブから掻き集めたテキストのソースとしてCommon Crawlを活用する。コモン

5. https://cloud.google.com/tpu/

Crawlは以前、NLPのテキストデータ源として、例えばn-gram言語モデルの訓練（Buckら、2014）、コモンセンス推論の訓練データ（Trinh and Le、2018）、機械翻訳用の並行テキストのマイニング（Smithら、2013）、事前訓練データセット（Graveら、2018；Zellersら、2019；Liuら、2019c）、さらには単にオプティマイザーのテスト用巨大テキストコーパス（Anilら、2019）として使用されています。

Common Crawlは、スクレイピングされたHTMLファイルからマークアップなどの非テキストコンテンツを除去し、「ウェブ抽出テキスト」を提供する一般公開のウェブアーカイブです。このプロセスにより、毎月約20TBのスクレイピングされたテキストデータが生成されます。しかし、残念ながら、抽出されたテキストの大半は、自然言語ではありません。その代わり、メニューやエラーメッセージ、重複したテキストなど、ちんぷんかんぷんなものや定型文が大半を占めています。さらに、スクレイピングされたテキストの多くには、私たちが考えるタスクに役立つとは思えないコンテンツ（不快な言葉、プレースホルダーテキスト、ソースコードなど）が含まれています。これらの問題に対処するため、Common Crawlのウェブ抽出テキストをクリーンアップするために、以下のヒューリスティックを使用しました：

- 末尾に句読点（ピリオド、エクスクラメーションマーク、クエスチョンマーク、エンドクォーテーションマークなど）がある行のみ保持しました。
- 5文未満のページは破棄し、3語以上の行を残すようにしました。
- 汚い言葉、いたずらな言葉、卑猥な言葉、その他悪い言葉のリスト」にある言葉を含むページを削除しました[6]。
- スクレイピングされたページの多くには、Javascriptを有効にする必要があるという警告が含まれていたため、Javascriptという言葉が含まれる行を削除しました。
- 一部のページで「lorem ipsum」のプレースホルダーがありましたが、「lorem ipsum」のフレーズがあるページを削除しました。
- 一部のページでコードが誤って表示されることがありました。中括弧「{」は、多くのプログラミング言語（ウェブで広く使われているJavascriptなど）で登場しますが、「{」は登場しません。

自然文の場合、中括弧を含むページを削除した。

- データセットを重複排除するために、データセット中に複数回出現する3文節のうち、1文節を除くすべてを破棄しました。

さらに、私たちの下流タスクのほとんどは英語のテキストに焦点を当てているため、langdetect[7]を使用して、少なくとも0.99の確率で英語として分類されていないページをフィルタリングしました。私たちのヒューリスティックは、Common CrawlをNLPのデータ源として使用する過去の研究に触発されています：例えば、Graveら（2018）も自動言語検出器を使用してテキストをフィルタリングし、短い行を破棄し、Smithら（2013）; Graveら（2018）は両方とも行レベルの重複排除を行っています。しかし、先行するデータセットがより限定的なフィルタリングヒューリスティックのセットを使用している、一般に公開されていない、および/または範囲が異なる（例えば、ニュースデータに限定されている（Zellers et al., 2019; Liu et al., 2019c）、クリエイティブコモンズのコンテンツのみからなる（Habernal et al., 2016）、機械翻訳用の並列トレーニングデータに焦点を当てている（Smith et al, 2013））ために、新しいデータを作ることを選択した。

6. https://github.com/LDNOOBW/List-of-Dirty-Naughty-Obscene-and-Otherwise-Bad-Words
7. https://pypi.org/project/langdetect/

ベースデータセットを組み立てるために、2019年4月のウェブ抽出テキストをダウンロードし、前述のフィルタリングを適用しました。これにより、事前学習に使用される多くのデータセットよりも桁違いに大きい（約750GB）だけでなく、適度にクリーンで自然な英語テキストからなるテキストのコレクションが生成されます。このデータセットを「Colossal Clean Crawled Corpus」（略してC4）と名付け、TensorFlow Datasetsの一部として公開します[8]。このデータセットのさまざまな代替バージョンを使用した場合の影響については、セクション3.4で検討します。

2.3. 下流タスク

本稿の目的は、一般的な言語学習能力を測定することである。そのため、機械翻訳、質問応答、抽象的要約、テキスト分類など、多様なベンチマークにおける下流の性能を研究しています。具体的には、GLUEおよびSuperGLUEテキスト分類メタベンチマーク、CNN/Daily Mail抽象的要約、SQuAD質問応答、WMT英語からドイツ語、フランス語、ルーマニア語翻訳に関するパフォーマンスを測定する。データはすべてTensorFlow Datasets.[9]から提供された。

GLUE (Wang et al, 2018) と SuperGLUE (Wang et al, 2019b) はそれぞれ、一般的な言語理解能力をテストするためのテキスト分類タスクのコレクションで構成されています：

- 文章受容性判定（CoLA（Warstadt et al, 2018））
- センチメント分析（SST-2（Socher et al, 2013））
- 言い換え・文の類似性（MRPC（Dolan and Brockett, 2005）、STS-B（Cer et al, 2017）、QQP（Iyer et al, 2017））
- 自然言語推論（MNLI（Williamsら、2017）、QNLI（Rajpurkarら、2016）、RTE（Daganら、2005）、CB（De Marneffら、2019））。
- 共参照解決（WNLIとWSC（Levesque et al, 2012）
- 文章補完（COPA（Roemmele et al, 2011））
- 語義曖昧性解消（WIC（Pilehvar and Camacho-Collados, 2018））
- 質問応答（MultiRC（Khashabi et al, 2018）、ReCoRD（Zhang et al, 2018）、

BoolQ（Clark et al, 2019)

GLUEおよびSuperGLUEベンチマークで配布されたデータセットを使用します。簡略化のため、微調整の際には、GLUEベンチマークのすべてのタスク（SuperGLUEについても同様）を、構成するすべてのデータセットを連結することで1つのタスクとして扱います。Kocijanら（2019）が示唆したように、SuperGLUEの結合タスクにはDefinite Pronoun Resolution（DPR）データセット（Rahman and Ng, 2012）も含まれます。

CNN/Daily Mail (Hermann et al., 2015) データセットは質問応答タスクとして導入されたが、Nallapati et al. (2016) によってテキスト要約に適応された。我々は抽象的要約タスクとして See et al. (2017) からの非匿名化バージョンを使用する。SQuAD (Rajpurkar et al., 2016) は一般的な質問応答ベンチマークである。我々の

8. https://www.tensorflow.org/datasets/catalog/c4
9. https://www.tensorflow.org/datasets

の実験では、モデルに質問とその文脈を与え、トークンごとに答えを生成するよう求めています。WMT英語からドイツ語については、（Vaswani et al., 2017）と同じトレーニングデータ（すなわち、News Commentary v13, Common Crawl, Europarl v7）と、検証セットとしてのnewstest2013（Bojar et al., 2014）を使用しています。英語からフランス語については、2015年の標準的なトレーニングデータと、検証セットとしてnewstest2014を使用します（Bojar et al., 2015）。標準的な低リソース機械翻訳ベンチマークである英語からルーマニア語については、WMT2016（Bojar et al.、2016）の訓練セットと検証セットを使用する。英語データでのみ事前学習を行うため、翻訳を学習するためには、与えられたモデルが新しい言語でテキストを生成することを学習する必要があることに注意してください。

2.4. 入出力フォーマット

つまり、モデルに文脈や条件付けのためのテキストを与え、その後に出力テキストを生成させるタスクである。このフレームワークは、事前学習と微調整の両方において、一貫した学習目的を提供します。具体的には、タスクに関係なく、モデルは最尤目標で訓練される（「教師強制」（Williams and Zipser, 1989）を使用）。モデルが実行すべきタスクを指定するために、モデルに入力する前に、元の入力シーケンスにタスク固有の（テキスト）プレフィックスを追加する。

例えば、"That is good. "という文章を英語からドイツ語に翻訳するようモデルに依頼する場合、モデルには "translate English to German: That is good. "というシーケンスが与えられ、"Das ist gut. "を出力するように学習される。テキストの分類タスクでは、モデルは単にターゲットラベルに対応する単一の単語を予測する。例えば、MNLIベンチマーク（Williams et al.、2017）では、前提が仮説を含意（「entailment」）、矛盾（「contradiction」）、またはどちらでもない（「neutral」）かを予測することが目標とされています。前処理を行うと、入力シーケンスは「mnli premise: I hate pigeons. hypothesis：仮説：私は鳩が嫌いである」となり、対応する目的語は「entailment」となる。なお、テキスト分類のタスクで、モデルが可能なラベルのいずれにも該当しないテキストを出力した場合（

例えば、タスクの可能なラベルが「entailment」「neutral」「contradiction」だけだった場合にモデルが「hamburger」と出力した場合）、問題が発生する。この場合、モデルの出力は常に間違いとしてカウントされますが、学習済みモデルでこのような挙動が見られたことはありません。あるタスクで使用されるテキストの接頭辞の選択は、本質的にハイパーパラメータであることに注意してください。接頭辞の正確な文言を変更しても影響は限定的であることがわかったため、接頭辞の選択に関する大規模な実験は行いませんでした。図1に、私たちのText-to-Textフレームワークの図と、いくつかの入力/出力例を示します。付録Dには、私たちが研究したすべてのタスクの前処理された入力の完全な例が示されています。

私たちのテキストトゥテキストフレームワークは、複数のNLPタスクを共通のフォーマットに落とし込む先行研究を踏襲しています：McCannら（2018）は、「Natural Language Decathlon」という、10個のNLPタスクのスイートに一貫した質問-回答形式を使用するベンチマークを提案しています。また、Natural Language Decathlonでは、すべてのモデルがマルチタスクであること、すなわち一度にすべてのタスクに同時に取り組むことができることが規定されています。我々はその代わりに、個々のタスクごとにモデルを個別に微調整できるようにし、明示的な質問と回答の形式ではなく、短いタスク接頭辞を使用します。Radfordら(2019)は、ゼロショット学習能力を評価しています。

言語モデルは、ある入力を前置詞としてモデルに与え、出力を自己回帰的にサンプリングすることによって行われます。例えば、「TL;DR:」（"too long, didn't read "の略、一般的な略語）の後に文書を入力し、自己回帰的にデコードすることで要約を予測することで、自動要約を行う。我々は主に、エンコーダで入力を明示的に処理してから別のデコーダで出力を生成するモデルを検討し、ゼロショット学習ではなく、転移学習に焦点を当てる。最後に、Keskarら（2019b）は多くのNLPタスクを「スパン抽出」として統一しており、可能な出力選択肢に対応するテキストを入力に付加し、正しい選択肢に対応する入力スパンを抽出するようにモデルを学習させる。これに対し、我々のフレームワークは、可能な出力選択肢をすべて列挙することができない、機械翻訳や抽象的要約のような生成的タスクも可能にする。

STS-Bは、1〜5の類似度スコアを予測する回帰タスクであるが、これを除いて、検討したタスクのすべてをテキストからテキストに変換することができた。これらのスコアのほとんどは0.2刻みでアノテーションされていることがわかったので、どのスコアも0.2刻みに丸め、その結果を文字列表現に変換した（例えば、浮動小数点値2.57は "2.6 "という文字列にマッピングされることになります）。テスト時には、モデルが1から5までの数字に対応する文字列を出力した場合、それを浮動小数点値に変換し、そうでない場合は、モデルの予測を不正解として扱います。これにより、STS-Bの回帰問題は21クラスの分類問題に生まれ変わる。

これとは別に、Winogradタスク（GLUEのWNLI、Super-GLUEのWSC、SuperGLUEに追加したDPRデータセット）を、よりシンプルな形式に変換し、テキスト-テキストフレームワークに適応させました。Winogradタスクの例では、文章中に複数の名詞句を指す可能性のある曖昧な代名詞が含まれる文章があります。例えば、"The city councilmen refused the demonstrators a permit because they fear violence." という文章があり、"they" という曖昧な代名詞が含まれていて、これは "city councilmen" または "demonstrators" を指すことができます。WNLI、WSC、DPRのタスクは、テキストからテキストへの問題として、テキスト文中の曖昧な代名詞をハイライトし、それが指す名詞を予測するようモデルに要求する。前述の例は、「The city councilmen refused the demonstrators a permit because *they* feared violence.」という

入力に変換され、モデルはターゲットテキスト「The city councilmen」を予測するように学習されることになる。

WSCでは、例文には、通路、曖昧な代名詞、候補となる名詞、候補が代名詞と一致するかどうかを反映する真偽ラベル（冠詞は無視）が含まれています。True」ラベルの例では、「False」ラベルの例では正しい名詞のターゲットが分からないため、「True」ラベルの例でのみ学習します。評価では、モデルの出力の単語が候補の名詞句の単語のサブセットである場合（またはその逆）、「True」ラベルを割り当て、それ以外の場合は「False」ラベルを割り当てます。これにより、WSCトレーニングセットの約半分が削除されましたが、DPRデータセットには約1,000個の代名詞解決例が追加されました。DPRの例には正しい参照名詞が注釈されているため、このデータセットを上記のフォーマットで簡単に使用することができます。

WNLIの訓練セットと検証セットは、WSCの訓練セットと大きく重複している。検証例が訓練データに漏れるのを避けるため（セクション3.5.2のマルチタスク実験では特に問題）、WNLIで訓練することはなく、WNLI検証セットでの結果も報告しないことにした。WNLI検証セットでの結果を省略することは、標準的な方法です。

の実践（Devlin et al., 2018）は、トレーニングセットに関して「敵対的」であるという事実、すなわち、検証例はすべて反対のラベルを持つトレーニング例のわずかに摂動されたバージョンであることに起因しています。そのため，検証セットについて報告するときは常に，WNLIをGLUEの平均スコアに含めない（テストセットで結果を示すセクション3.7を除くすべてのセクション）．WNLIから上記の「参照名詞予測」への変換はもう少し複雑で、このプロセスについては付録Bで説明します。

3. 実験風景

NLPのための転移学習の最近の進歩は、新しい事前学習目的、モデルアーキテクチャ、ラベルなしデータセットなど、様々な開発から生まれている。本節では、これらの技術の実証的な調査を行い、その貢献度と重要性を明らかにすることを目的とする。そして、得られた知見を組み合わせて、我々が検討する多くのタスクで最先端を達成する。NLPのための転移学習は急速に発展している研究分野であるため、私たちの実証的な研究において、ありとあらゆる技術やアイデアを網羅することは現実的ではありません。より広範な文献レビューについては、Ruderら（2019）による最近のサーベイを推奨します。

我々は、合理的なベースライン（セクション3.1で説明）を取り、一度にセットアップの1つの側面を変更することにより、これらの貢献を体系的に研究する。例えば、セクション3.3では、実験パイプラインの残りの部分を固定したまま、異なる教師なし目的のパフォーマンスを測定する。この「座標上昇」アプローチは、2次的な効果（例えば、ある特定の教師なし目的は、ベースライン設定よりも大きなモデルで最も効果的かもしれない）を見逃すかもしれないが、我々の研究ですべての要因の組み合わせ的な調査を行うことは、法外な費用がかかる。将来的には、本研究で検討したアプローチの組み合わせをより徹底的に検討することが実りあるものになると期待される。

私たちの目標は、できるだけ多くの要素を固定したまま、多様なタスクセットで様々な異なるアプローチを比較することです。この目的を満たすために、場合によっては、既存のアプローチを正確に再現しないこともあります。例えば、BERT（Devlin et al., 2018）のような「エンコーダのみ」のモデルは、入力トークンごとに単一の予測を行うか、入力シーケンス全体に対して単一の予測を行うように設計されています。このため、分類やスパン予測タスクには適用できますが、翻訳や抽象的要約のような生成的タスクには適用できません。このように、私たちが検討したモデル・アーキテクチャは、いずれもBERTと同一ではなく、エンコーダのみの構造で構成されています。例えば、セクション 3.3 では、BERT の「マスクされた言語モデリング」目的に類似した目的を検討し、セクション 3.2 では、テキスト分類タスクで BERT に類似した動作をするモデルアーキテクチャを検討することで、その代わりに、精神的に類似したアプローチをテストしています。

次のサブセクションでベースラインの実験設定を概説した後、モデルアーキテクチャ（セクション3.2）、教師なし目的（セクション3.3）、事前学習データセット（セクション3.4）、転送アプローチ（セクション3.5）、スケーリング（セクション3.6）の実証比較を実施する。本節の最後に、本研究で得られた知見とスケールを組み合わせることで、我々が考える多くのタスクで最先端の結果を得ることができる（セクション3.7）。

3.1. ベースライン

私たちのベースラインの目標は、典型的な現代の実践を反映することです。我々は、単純なノイズ除去目標を用いて標準的なTransformer（セクション2.1で説明）を事前に訓練し、その後、下流の各タスクで別々に微調整を行う。この実験セットアップの詳細については、以下のサブセクションで説明する。

3.1.1. モデル

我々のモデルには、Vaswaniら（2017）が提案したような標準的なエンコーダ・デコーダTransformerを使用する。NLPのための転移学習に対する多くの最新のアプローチは、単一の「スタック」のみからなるTransformerアーキテクチャを使用するが（例えば、言語モデリング（Radford et al., 2018; Dong et al., 2019）または分類とスパン予測（Devlin et al., 2018; Yang et al., 2019））、我々は標準エンコーダ-デコーダ構造の使用は生成および分類タスク両方で良い結果を達成したことがわかった。セクション3.2で、異なるモデルアーキテクチャの性能を調査する。

私たちのベースラインモデルは、エンコーダとデコーダがそれぞれ「$\text{BERT}_{\text{BASE}}$」（Devlin et al., 2018）スタックに似たサイズと構成になるように設計されています。具体的には、エンコーダとデコーダの両方が12個のブロック（各ブロックは、自己注意、オプションのエンコーダ-デコーダ注意、およびフィードフォワードネットワークで構成される）で構成されています。各ブロックのフィードフォワードネットワークは、出力次元が$d_{\text{ff}} = 3072$の密な層と、それに続くReLU非線形と別の密な層で構成されています。すべての注意メカニズムの「キー」と「値」の行列は、$d_{\text{kv}} = 64$の内部次元を持ち、すべての注意メカニズムは12個のヘッドを持っている。その他のすべてのサブレイヤーとエンベッディングの次元は$d_{\text{model}} = 768$である。この結果、合計で約2億2千万個のパラメータを持つモデルになりました。これは、$\text{BERT}_{\text{BASE}}$ のパラメータ数のおよそ2倍である。これは、我々のベースラインモデルが1層ではなく、2層のスタックを含んでいるからである。正則化については、モデル内でドロップアウトが適用されているすべての場所で、ドロップアウト確率 0.1 を使用しています。

3.1.2. トレーニング

セクション2.4で述べたように、すべてのタスクはテキストからテキストへのタスクとして定式化されている。これにより、常に標準的な最尤法を用いて、すなわち教師強制（Williams and Zipser, 1989）とクロスエントロピー損失を用いて訓練することができる。最適化には、AdaFactor (Shazeer and Stern, 2018)を使用する。テスト時には、貪欲なデコード（すなわち、タイムステップごとに最高確率のロジットを選択する）を使用する。

各モデルをC4で2^{19}=524,288ステップの事前学習を行い、微調整を行う。最大配列長を512とし、バッチサイズを128配列とする。可能な限り、複数のシーケンスをバッチの各エントリーに「詰め込む」[10] ので、バッチにはおよそ $2^{16} = 65{,}536$ 個のトークンが含まれる。このバッチサイズとステップ数を合計すると、以下のようになる。
を2^{35}≒34Bトークンで事前学習する。これは、およそ137Bトークンを使用したBERT（Devlin et al.、2018）や、およそ
2.2Tトークンです。2^{35}個のトークンのみを使用することで、合理的な計算予算が得られる一方、十分な量の事前学習が行われ、許容できる性能を得ることができます。を検討する。

10. https://www.pydoc.io/pypi/tensor2tensor-1.5.7/autoapi/data_generators/generator_utils/index.html#data_generators.generator_utils.pack_examples

の効果は、セクション3.6と3.7でより多くのステップでプリトレーニングを行うことができます。235個のトークンはC4データセットのほんの一部に過ぎないため、事前学習中にデータを繰り返すことはないことに注意してください。

事前学習では、「逆平方根」の学習率スケジュールを使用する：$1/\sqrt{\max(n, k)}$ ここで、nは現在の学習反復、kはウォームアップステップの数である（すべての実験で104に設定）。これにより、最初の104ステップでは0.01の一定の学習率が設定され、その後、事前学習が終了するまで学習率が指数関数的に減衰します。また、三角形の学習率（Howard and Ruder, 2018）を使用する実験も行いましたが、これはわずかに良い結果をもたらしましたが、事前にトレーニングステップの総数を知っておく必要があります。実験の一部で学習ステップ数を変化させる予定なので、より一般的な逆平方根のスケジュールを選択しました。

我々のモデルは、すべてのタスクで $2^{18} = 262{,}144$ ステップでファインチューニングされています。この値は、微調整の恩恵を受ける高リソースタスク（大規模なデータセットを持つタスク）と、すぐにオーバーフィットする低リソースタスク（小規模なデータセット）の間のトレードオフとして選択された。微調整の間、128長さ512のシーケンス（つまりバッチあたり216トークン）のバッチを使い続ける。微調整の際には、0.001の一定の学習率を使用する。5,000ステップごとにチェックポイントを保存し、最も高い検証性能に対応するモデルのチェックポイントで結果を報告する。複数のタスクでファインチューニングされたモデルについては、各タスクで独立に最適なチェックポイントを選択する。セクション3.7を除く全ての実験において、テストセットでのモデル選択を避けるため、検証セットでの結果を報告する。

3.1.3. ボキャブラリー

SentencePiece (Kudo and Richardson, 2018) を使用して、テキストを WordPiece トークン (Sennrich et al, 2015; Kudo, 2018) としてエンコードします。すべての実験では、32,000ワードピースの語彙を使用します。我々は最終的に英語からドイツ語、フランス語、ルーマニア語への翻訳で我々のモデルを微調整するので、我々の語彙がこれらの非英語をカバーすることも要求される。そこで、C4で使用した

Common Crawlスクレイプからページをドイツ語、フランス語、ルーマニア語に分類しました。そして、英語のC4データ10部と、ドイツ語、フランス語、ルーマニア語に分類されたデータ各1部を混合して、SentencePieceモデルを学習しました。この語彙は、モデルの入力と出力の両方で共有されました。この語彙のおかげで、モデルはあらかじめ決められた固定された言語セットしか処理できないことに注意してください。

3.1.4. 非教師的目的

ラベルのないデータを活用してモデルを事前学習させるためには、ラベルを必要としないが、（緩やかに言えば）下流のタスクで有用な一般化可能な知識をモデルに教えるという目的が必要である。モデルの全パラメータを事前学習して微調整するという転移学習のパラダイムをNLP問題に適用した予備的な研究では、事前学習に因果関係言語モデリング目的を使用していました（Dai and Le, 2015; Peters et al., 2018; Radford et al., 2018; Howard and Ruder, 2018）。しかし、最近、「ノイズ除去」目的（Devlin et al., 2018; Taylor, 1953）（「マスクされた言語モデリング」とも呼ばれる）がより良いパフォーマンスを生み出すことが示され、その結果、急速に標準となりました。ノイズ除去目的では、モデルは、入力中の欠損またはその他の破損したトークンを予測するように訓練されます。BERTの「マスクド・ランゲージ・モデリング」目標と、「マスクド・ランゲージ・モデリング」目標に触発された

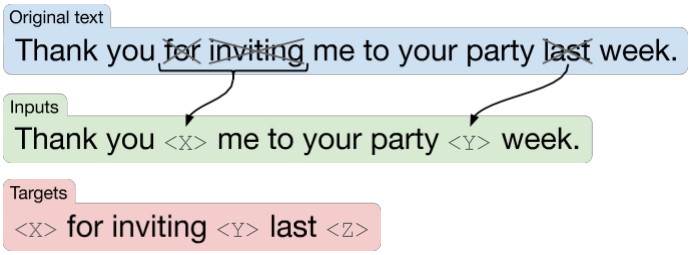

図2：ベースラインモデルで使用している目的語の模式図。この例では、"Thank you for inviting me to your party last week. "という文章を処理します。その際
for"、"inviting"、"last"（×印）の単語がランダムに選ばれ、破損される。破損したトークンの連続するスパンは、それぞれセンチネルに置き換えられる。
トークン（<X>と<Y>で示される）は、例文の上で一意である。for」と「inviting」は連続して出現するため、1つのセンチネル<X>で置き換えられる。出力シーケンスは、ドロップアウトしたスパンと、入力で置換に使われたセンチネルトークンと最後のセンチネルトークン<Z>で区切られたものである。

"word dropout "正則化手法（Bowman et al, 2015）では、入力シーケンスのトークンの15%をランダムにサンプリングし、その後ドロップアウトする目的を設計します。ドロップアウトされたトークンの連続するスパンはすべて、単一のセンチネルトークンに置き換えられる。各センチネルトークンには、シーケンスに固有なトークンIDが割り当てられる。センチネルIDは語彙に追加される特別なトークンであり、どの単語にも対応しない。ターゲットは、入力シーケンスで使用されたのと同じセンチネルトークンと、ターゲットシーケンスの終わりを示す最後のセンチネルトークンで区切られた、脱落したトークンのすべてのスパンに対応します。連続するトークンのスパンをマスクし、脱落したトークンのみを予測するのは、事前学習の計算コストを削減するためである。プリトレーニングの目的については、セクション3.3で徹底的に調査する。この目的を適用した結果の変換の例を図2に示す。3.3節では、この目的語を他の多くの変化形と経

験的に比較する。

3.1.5. ベースライン・パフォーマンス

このセクションでは、上述したベースライン実験手順を用いた結果を示し、一連の下流タスクでどのようなパフォーマンスが期待できるかを把握することができます。理想的には、本研究のすべての実験を複数回繰り返し、結果の信頼区間を得ることです。しかし、実験回数が多いため、この方法は非常に高価になります。安価な代替案として、ベースラインモデルをゼロから10回訓練し（すなわち、異なるランダムな初期化とデータセットのシャッフルを用いて）、ベースモデルのこれらの実行における分散が、各実験バリエーションにも適用されると仮定する。私たちが行う変更のほとんどは、実行間分散に劇的な影響を与えるとは思わないので、これによって、異なる変更の重要性を合理的に示すことができるはずです。また、これとは別に、事前学習なしで、すべての下流タスクに対して2^{18}ステップ（微調整に使用するのと同じ数）のモデル学習を行った場合のパフォーマンスも測定しています。これにより、ベースライン設定において、事前学習がモデルにどの程度の利益をもたらすかを知ることができます。

	グリュー	シーエヌエヌエム	スクワッド	スグレもの	エンデ	EnFr	EnRo
F ベースライン平均	**83.28**	**19.24**	**80.88**	**71.36**	**26.98**	**39.82**	**27.65**
ベースラインの標準偏差	0.235	0.065	0.343	0.416	0.112	0.090	0.108
事前トレーニングなし	66.22	17.60	50.31	53.04	25.86	**39.77**	24.04

表1：ベースラインモデルとトレーニング手順で達成したスコアの平均と標準偏差。比較のため、ベースラインモデルの微調整に使用したのと同じステップ数で、各タスクをゼロから（つまり事前訓練なしで）訓練した場合のパフォーマンスも報告している。この表（および表14を除くすべての表）のスコアは、各データセットの検証セットで報告されています。

本文で結果を報告する際には、スペースを節約し、解釈を容易にするために、すべてのベンチマークのスコアのサブセットのみを報告する。GLUEとSuperGLUEについては、「GLUE」と「SGLUE」の見出しで、（公式ベンチマークで規定されている）すべてのサブタスクの平均スコアを報告する。すべての翻訳タスクについて、SacreBLEU v1.3.0 (Post, 2018)が提供するBLEUスコア（Papineni et al., 2002）を、「exp」スムージングと「intl」トークン化で報告する。WMT English to German、English to French、English to Romanianのスコアをそれぞれ EnDe、EnFr、EnRoと呼ぶことにする。CNN/Daily Mailについては、ROUGE-1-F、ROUGE-2-F、ROUGE-L-F (Lin, 2004)の各メトリクスにおけるモデルの性能に高い相関があることがわかったので、ROUGE-2-Fスコアのみを「CNNDM」の見出しで報告します。同様に、SQuADについては、「完全一致」スコアと「F1」スコアの性能に高い相関があることがわかったので、「完全一致」スコアのみを報告します。すべての実験において、すべてのタスクで達成されたスコアは、付録Eの表16に記載されています。

結果の表はすべて、各行が特定の実験構成に対応し、列が各ベンチマークのスコアを示すようにフォーマットされています。ほとんどの表で、ベースライン構成の平均性能を記載しています。ベースライン構成が登場する場合は、（表1の1行目のように）Fでマークします。また、ある実験での最大値（最高値）から2標準偏差以

内のスコアは**太字で表示することにする。**

我々のベースライン結果を表1に示す。全体として、我々の結果は、同規模の既存モデルと同等である。例えば、$BERT_{BASE}$ は、SQuAD で 80.8 の完全一致スコアと MNLI-matched で 84.4 の精度を達成しましたが、我々はそれぞれ 80.88 と 84.24 を達成しました（表 16 参照）。私たちのベースラインを直接比較することはできないことに注意してください。

$BERT_{BASE}$ エンコーダー・デコーダーモデルであり、およそ事前学習されているためです。
を何段階にも分けて行うことができます。当然のことながら、事前トレーニングによって、以下のような大きな効果が得られることがわかりました。
ほぼすべてのベンチマークで唯一の例外はWMT英仏で、これは十分に大きなデータセットであるため、事前学習による利益はわずかなものになりがちである。このタスクは、高リソース領域における転移学習の挙動を検証するために実験に組み込んだものである。最も成績の良いチェックポイントを選択して早期停止を行うため、ベースラインと「事前学習なし」の間の大きな格差は、データが限られたタスクにおいて事前学習がどれほど成績を向上させるかを強調している。本稿では、データ効率の向上を明示的に測定していないが、これは転移学習パラダイムの主要な利点の1つであることを強調している。

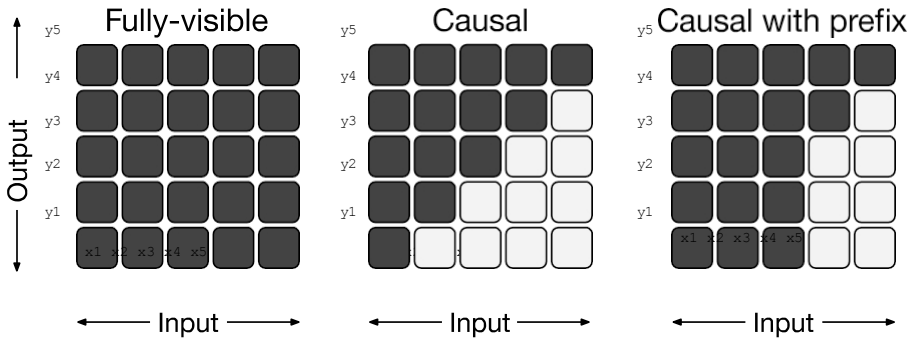

図3：異なるアテンションマスクパターンを表すマトリックス。自己注意メカニズムの入力と出力はそれぞれxと*y*と表記される。行iと列*j*にある暗いセルは、出力タイムステップ*i*で自己注意メカニズムが入力要素*j*に注意することを許可されていることを示し、明るいセルは、自己注意メカニズムが対応するiと*j*の組み合わせに注意することを許可されていないことを示します。左：完全に見えるマスクは、自己留保メカニズムがすべての出力タイムステップで全入力に出席することを許可する。真ん中：因果関係のあるマスクは、i番目の出力要素が「未来」からの入力要素に依存することを防ぐ。右：接頭辞を持つ因果的マスクは、自己注意機構が入力シーケンスの一部に対して完全に可視化されたマスクを使用することを可能にします。

ラン間の分散については、ほとんどのタスクで、ラン間の標準偏差は、そのタスクのベースライン・スコアの1%より小さいことがわかります。例外として、CoLA、CB、COPAはGLUEとSuperGLUEベンチマークの低リソースタスクである。例えば、CBでは、ベースラインモデルの平均F1スコアは91.22、標準偏差は3.237（表16参照）でしたが、これはCBの検証セットには56例しか含まれていないことが一因かもしれません。なお、GLUEとSuperGLUEのスコアは、各ベンチマークを構成するタスクのスコアの平均値として計算されている。そのため、CoLA、CB、COPAの実行間分散が大きいため、GLUEとSuperGLUEのスコアだけではモデルの比較が難し

くなる可能性があることに注意してください。

3.2. アーキテクチャー

Transformerはもともとエンコーダとデコーダのアーキテクチャで導入されましたが、NLPのための転移学習に関する現代の研究の多くは、別のアーキテクチャを使用しています。このセクションでは、これらのアーキテクチャのバリエーションについてレビューし、比較する。

3.2.1. モデル構造

異なるアーキテクチャを区別する大きな要因は、モデル内の異なる注意メカニズムが使用する「マスク」である。Transformerの自己注意操作は、入力としてシーケンスを受け取り、同じ長さの新しいシーケンスを出力することを思い出してください。出力シーケンスの各エントリーは、入力シーケンスのエントリーの加重平均を計算することによって生成される。具体的には、y_i を出力シーケンスの第 1 要素とし、x_j を

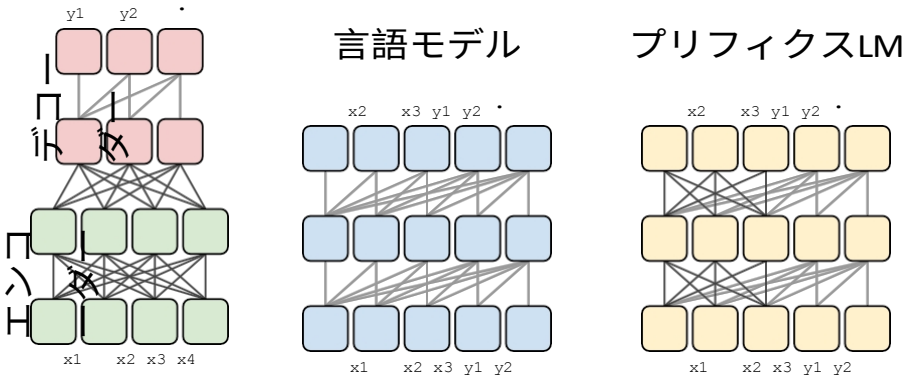

図4：我々が検討するTransformerアーキテクチャのバリアントの回路図。この図において、ブロックはシーケンスの要素を表し、線は注目の可視性を表す。異なる色のブロック群は、異なるTransformerレイヤースタックを示す。濃いグレーの線は完全可視マスキングに対応し、薄いグレーの線は因果的マスキングに対応する。予測値の終わりを表す特別なend-of-sequenceトークンを表すために". "を使用する。入力と出力のシーケンスは、それぞれxとyで表される。左：標準的なエンコーダ-デコーダアーキテクチャでは、エンコーダとエンコーダ-デコーダのアテンションで完全可視マスキングを使用し、デコーダで因果マスキングを使用します。真ん中：言語モデルは1つのTransformer層スタックで構成され、入力とターゲットの連結が与えられ、全体的に因果的なマスクが使用される。右：言語モデルにプレフィックスを追加することで、入力に対して完全に可視化されたマスキングを可能にすることに相当する。

y_i は，$\Sigma_j w\, x_{i,jj}$ として計算される。ここで，$w_{i,j}$ は，x_i と x_j の関数として，自己注意メカニズムが生成するスカラー重みである。次に，注意マスクは，入力シーケンスのどのエントリを制約するために，特定の重みをゼロにするために使われる。
は、与えられた出力タイムステップにおいて注目されることができる。図3は、これから検討するマスクの図である。例えば、因果関係マスク（図3、中）は、$j > i$の場合、任意の$w_{i,j}$ をゼロに設定する。

最初に考えるモデル構造は、2つのレイヤの限界に対処するエンコーダ・デコーダ・トランスフォーマーである：エンコーダーは入力シーケンスが供給され、デコーダーは新しい出力シーケンスを生成する。このアーキテクチャの概略図を図4の左パネルに示す。

エンコーダは「完全可視」アテンションマスクを使用する。完全可視化マスクは、自己の注意メカニズムが、その出力の各項目を生成する際に、入力のどの項目にも注意することを可能にします。このマスキングパターンを図3（左）に示します。この形式のマスキングは、「接頭辞」、すなわち、後に予測を行う際に使用される、モデルに提供される何らかのコンテキストに注目する場合に適切である。BERT（Devlinら、2018）も完全に可視化されたマスキングパターンを使用し、入力に特別な「分類」トークンを付加します。そして、分類トークンに対応するタイムステップでのBERTの出力は、入力シーケンスを分類するための予測を行うために使用されます。

Transformerのデコーダにおける自己注意操作は、「因果的」なマスキングパターンを使用しています。これは、出力シーケンスの*i番目の*エントリーを生成する際に、モデルが入力シーケンスの*j番目の*エントリーに注目しないようにするもので、$j > i$の場合、学習時に使用します。このマスキング・パターンのアテンション・マトリックスを図3の中段に示す。

エンコーダ・デコーダTransformerのデコーダは、出力シーケンスを自己回帰的に生成するために使用されます。つまり、各出力タイムステップで、トークンがモデルの予測分布からサンプリングされ、そのサンプルが次の出力タイムステップの予測を生成するためにモデルにフィードバックされる、といった具合である。このように、Transformerデコーダ（エンコーダなし）は、言語モデル（LM）、すなわち次ステップ予測のためだけに訓練されたモデルとして使用できる（Liuら、2018；Radfordら、2018；Al-RFOUら、2019）。これは、我々が考える第2のモデル構造を構成する。このアーキテクチャの模式図を図4中段に示す。実際、NLPのための転移学習に関する初期の研究では、事前学習法として言語モデリングを目的としたこのアーキテクチャを使用していました（Radford et al.、2018）。

言語モデルは通常、圧縮や配列生成に使用される（Graves, 2013）。しかし、言語モデルは、入力とターゲットを連結するだけで、text-to-textフレームワークで使用することもできる。例として、英語からドイツ語への翻訳を考えてみましょう：入力文 "That is good. "とターゲット "Das ist gut. "を持つトレーニングデータポイントがあれば、連結された入力列 "translate English to German: That is good. target "に対して次のステップの予測モデルをトレーニングするだけです：Das ist gut."この例に対するモデルの予測を得たい場合、モデルには "translate English to German: That is good. target: "という接頭辞が与えられ、残りのシーケンスを自己回帰的に生成するよう求められる。このように、モデルは入力があれば出力シーケンスを予測することができ、テキスト-テキストタスクのニーズを満たすことができる。このアプローチは、最近、言語モデルが監督なしでいくつかのtext-to-textタスクを実行することを学習できることを示すために使用された（Radford et al.、2019）。

これがなぜ潜在的に不利なのかを知るために、モデルに予測をさせる前に接頭

辞/文脈を提供する(例えば、接頭辞は英語の文章で、モデルはドイツ語訳を予測するよう求められる)テキスト-to-テキストの枠組みを考えてみましょう。完全因果マスキングでは、モデルによる接頭辞の状態の表現は、接頭辞の先行エントリにのみ依存することができます。そのため、出力のエントリーを予測する際、モデルは不必要に制限された接頭辞の表現に注目することになる。同様の議論は、sequence-to-sequenceモデルで一方向性のリカレントニューラルネットワークエンコーダを使用することに対して行われている（Bahdanau et al.、2015）。

この問題は、Transformerベースの言語モデルにおいて、単にマスキングパターンを変更することで回避することができる。因果関係のあるマスクの代わりに、シーケンスのプリフィックス部分において完全に可視化されたマスクを使用するのである。このマスキングパターンと、その結果得られる「接頭辞LM」（我々が考える3番目のモデル構造）の概略を、それぞれ図3および図4の右端のパネルに示している。前述の英独翻訳の例では、"translate English to German: That is good. target: "という接頭辞には完全可視マスキングを適用し、"Das ist gut "というターゲットを予測する訓練には因果マスキングを適用した。テキスト・トゥ・テキストフレームワークで接頭辞LMを使うことは、元々、以下のように提案された。

Liuら(2018)による。さらに最近、Dongら（2019）は、このアーキテクチャが多種多様なテキストトゥテキストタスクで有効であることを示した。このアーキテクチャは、エンコーダとデコーダに渡ってパラメータが共有され、エンコーダとデコーダの注意が入力とターゲットシーケンスに渡る完全な注意に置き換えられたエンコーダデコーダモデルに似ています。

我々のテキストトゥテキストフレームワークに従うと、接頭辞LMアーキテクチャは、分類タスクのBERT（Devlin et al, 2018）に酷似していることに注意する。その理由を知るために、MNLIベンチマークから、前提が "I hate pigeons."、仮説が "My feelings towards pigeons are filled with animosity."、正しいラベルが "entailment "という例を考えてみましょう。この例を言語モデルに投入するには、「mnli premise: I hate pigeons. hypothesis：仮説：私は鳩が嫌いである仮説：鳩に対する私の感情は敵意で満たされている。この場合、完全に可視化された接頭辞は、「target:」という単語までの入力シーケンス全体に対応することになり、これはBERTで使用される「分類」トークンに類似していると見なすことができる。つまり、このモデルは、入力全体を完全に可視化し、「entailment」という単語を出力することによって分類を行うことを任務とすることになる。タスクの接頭辞（この場合は「mnli」）が与えられた有効なクラスラベルのいずれかを出力するように、モデルが学習するのは簡単です。このように、プリフィックスLMとBERTアーキテクチャの主な違いは、プリフィックスLMでは分類器がTransformerデコーダの出力層に統合されるだけである。

3.2.2. 異なるモデルの構造を比較する

これらのアーキテクチャを実験的に比較するためには、検討する各モデルが何らかの意味で同等であることが望まれます。例えば、2つのモデルが同じ数のパラメータを持つか、与えられた（入力-配列、ターゲット-配列）ペアを処理するのに必要な計算量がほぼ同じであれば、2つのモデルは同等であると言うことができる。残念ながら、エンコーダー・デコーダーモデルと言語モデルアーキテクチャ（単一のTransformerスタックからなる）を、これらの基準の両方に従って同時に比較することは不可能です。その理由は、まず、エンコーダーにL層、デ

コーダーにL層を持つエンコーダー・デコーダーモデルが、2L層を持つ言語モデルとほぼ同じ数のパラメータを持つことに注目する。しかし、同じ$L + L$のエンコーダ・デコーダモデルは、L層しかない言語モデルとほぼ同じ計算量になります。これは、言語モデルのL層が入力配列と出力配列の*両方に*適用されなければならないのに対し、エンコーダは入力配列にのみ、デコーダは出力配列にのみ適用されるという事実の結果である。エンコーダとデコーダの注意によってデコーダには余分なパラメータがあり、注意層にはシーケンス長に二次関数をかける計算コストがかかるからだ。しかし、実際には、L*層の*言語モデルと$L + L$層のエンコーダ・デコーダモデルのステップ時間はほぼ同じであり、ほぼ同等の計算コストであることが示唆されました。さらに、今回検討したモデルサイズでは、エンコーダ・デコーダの注目層のパラメータ数は全パラメータ数の約10%であるため、$L + L$*層の*エンコーダ・デコーダモデルは2L層の言語モデルと同じパラメータ数であると単純化して仮定する。

合理的な比較手段を提供するため、エンコーダ・デコーダモデルについて複数の構成を検討する。$\text{BERT}_{\text{BASE}}$ サイズのレイヤースタックにおけるレイヤーの数とパラメータの数を、それぞれ L および P と呼ぶことにする。の数を示すために、M を使用する。

*L+L層の*エンコーダーデコーダーモデルと*L層の*デコーダーオンリーモデルが、与えられた入力とターゲットのペアを処理するために必要なFLOPsの比較。合計で、比較することになります：

- エンコーダに*L*層、デコーダに*L*層を持つエンコーダ・デコーダモデル。このモデルは2*P*個のパラメータを持ち、計算コストは*M* FLOPsである。
- 同等のモデルだが、エンコーダとデコーダでパラメータを共有するため、パラメータは*P個*、計算量は*M* -FLOPとなる。
- エンコーダとデコーダにそれぞれ*L*/2層を持つエンコーダ・デコーダモデルであり、*P*パラメータと*M/2*-FLOPコストで構成されています。
- *L*層、*P個の*パラメータを持つデコーダのみの言語モデルで、計算量は*M* FLOPsとなる。
- デコーダのみのプレフィックスLMは、同じアーキテクチャ（したがって、同じパラメータ数と計算コスト）でありながら、完全に可視化された自己アテンションを持つ。

  を入力します。

3.2.3. 目的

教師なし目的としては、基本的な言語モデリング目的と、セクション3.1.4で説明したベースラインのノイズ除去目的の両方を検討します。言語モデリング目的は、事前学習目的としての歴史的な使用（Dai and Le, 2015; Ramachandran et al., 2016; Howard and Ruder, 2018; Radford et al., 2018; Peters et al., 2018）と、我々が考える言語モデルアーキテクチャに自然に適合することから含める。予測を行う前に接頭辞を取り込むモデル（エンコーダーデコーダーモデルと接頭辞LM）については、ラベルなしデータセットからテキストのスパンをサンプリングし、接頭辞部分とターゲット部分に分割するランダムポイントを選択します。標準言語モデルについては、最初から最後までのスパン全体を予測するようにモデルを訓練します。教師なしノイズ除去の目的は、テキストからテキストへのモデル用に設計されています。言語モデルで使用するために、セクション3.2.1で説明したように、入力とターゲットを連結させます。

3.2.4. 結果

比較した各アーキテクチャが達成したスコアを表2に示します。すべてのタスクにおいて、ノイズ除去を目的としたエンコーダーデコーダーアーキテクチャが最も良い結果を示しました。このアーキテクチャは、パラメータ数が最も多い（2P）が、P*個の*パラメータを持つデコーダのみのモデルと同じ計算コストである。意外なことに、エンコーダーとデコーダーでパラメータを共有しても、ほぼ同じ結果が得られることがわかりました。一方、エンコーダとデコーダのスタックの層数を半分にすると、性能が著しく低下することがわかりました。同時進行の研究（Lan et al., 2019）も、Transformerブロック間でパラメータを共有することが、性能をあまり犠牲にすることなく、パラメータの総数を減らす有効な手段になり得ることを発見しました。XLNetは、ノイズ除去を目的とした共有エンコーダ・デコーダアプローチ（Yang et al.、2019）とも類似している。また、共有パラメータエンコーダ-デコーダがデコーダのみのプレフィックスLMを上回ることに注目し、明示的なエンコーダ-デコーダの注意を追加することが有益であることを示唆する。最後に、ノイズ除去目的語を使用すると、言語と比較して常に下流タスクのパフォーマンスが向上するという広く知られている概念を確認します。

建築	目的	パラム	コスト	グリュー	シーエヌエヌエム	スクワッド	スグレもの	エンデ	エヌファール	エンロ
F エンコーダ・デコーダ	デノイズ	*2P*	*M*	**83.28**	**19.24**	**80.88**	**71.36**	**26.98**	**39.82**	**27.65**
Enc-dec、共有	デノイズ	*P*	*M*	82.81	18.78	**80.63**	**70.73**	26.72	39.03	**27.46**
Enc-dec、6層	デノイズ	*P*	M/2	80.88	18.97	77.59	68.42	26.38	38.40	26.95
言語モデル	デノイズ	*P*	*M*	74.70	17.93	61.14	55.02	25.09	35.28	25.86
プリフィクスLM	デノイズ	*P*	*M*	81.82	18.61	78.94	68.11	26.43	37.98	27.39
エンコーダ・デコーダ	エルエム	*2P*	*M*	79.56	18.59	76.02	64.29	26.27	39.17	26.86
Enc-dec、共有	エルエム	*P*	*M*	79.60	18.13	76.35	63.50	26.62	39.17	27.05
Enc-dec、6層	エルエム	*P*	M/2	78.67	18.26	75.32	64.06	26.13	38.42	26.89
言語モデル	エルエム	*P*	*M*	73.78	17.54	53.81	56.51	25.23	34.31	25.38
プリフィクスLM	エルエム	*P*	*M*	79.68	17.84	76.87	64.86	26.28	37.51	26.76

表 2：セクション 3.2.2 で説明したさまざまなアーキテクチャの性能。*P* は 12 層ベーストランスフォーマーレイヤースタックのパラメータ数、*M* はエンコーダーデコーダーモデルを使用してシーケンスを処理するために必要な FLOPs を意味するものとして使用する。我々は、ノイズ除去目的（セクション 3.1.4 で説明）と自己回帰目的（言語モデルの訓練に一般的に使用される）を使用して、各アーキテクチャのバリエーションを評価する。

をモデリングすることを目的としています。この観察は、特にDevlinら（2018）、Voitaら（2019）、Lample and Conneau（2019）によって以前になされたものである。我々は、次のセクションで教師なし目的のより詳細な探索を行う。

3.3. 教師なしオブジェクト

教師なし目的の選択は、モデルが下流のタスクに適用するための汎用的な知識を獲得するメカニズムを提供するため、中心的な重要性を持っています。このため、多種多様な事前訓練目的が開発されている（Dai and Le, 2015; Ramachandran et al., 2016; Radford et al., 2018; Devlin et al., 2018; Yang et al., 2019; Liu et al., 2019b; Wang et al., 2019a; Song et al., 2019; Dong et al., 2019; Joshi et al., 2019）。このセクションでは、教師なし目的の空間の手続き的な探索を行う。多くの場合、我々は既存の目的を正確に複製するのではなく、いくつかは我々のテキスト-テキストエンコーダ-デコーダ

のフレームワークに合うように修正され、他のケースでは、複数の一般的なアプローチからの概念を組み合わせた目的を使用する。

全体として、すべての目的は、ラベルのないテキストデータセットから、トークン化されたテキストのスパンに対応するトークンIDのシーケンスを取り込みます。トークン列は処理され、（破損した）入力列と対応するターゲットが生成される。そして、モデルは通常通り最尤法で学習され、ターゲットシーケンスを予測する。表3は、私たちが考える多くの目的の例である。

3.3.1. バラバラのハイレベルなアプローチ

まずはじめに、一般的に使用されている目的にヒントを得ているが、そのアプローチが大きく異なる3つの手法を比較する。まず、3.2.3節で使用したような基本的な「前置詞の言語モデリング」目的を含めることにする。この手法は、テキストを2つの要素に分割し、一方をエンコーダの入力として、もう一方をターゲットシーケンスとして使用するものである。

目的	インプット	ターゲット
接頭辞言語モデリング	ご招待ありがとうございました。	先週、あなたのパーティーに参加しました。
BERT-style Devlinら（2018年）。	ありがとうございます<m> <m>あなたのパーティーのアップルウィークに私を .	*(原文)*
デスハフリング	.最後の楽しみのために、あなたのために、あなたの招待する週のためにパーティーをする。	*(原文)*
MASS式 Songら(2019)	ありがとうございます！<M> <M>あなたのパーティーへ <M>週 .	*(原文)*
I.I.D.ノイズ、スパンの交換	ありがとうございます<X>あなたのパーティに<Y>週。	<X>を招待するため <Y>を最後に <Z>を招待するため
I.I.D.ノイズ、ドロップトークン	あなたのパーティーウィークに私をありがとうございます。	さかんに誘う
ランダムスパン	ありがとうございます<X>から<Y>週へ.	<X> <Y> <Z>のパーティに招待してくれたこと。

表3：入力テキスト "Thank you for inviting me to your party last week ." に適用した教師なし目標のいくつかによって生成される入力と目標の例。我々の目的はすべて*トークン化された*テキストを処理していることに注意してください。この特定の文では、すべての単語が語彙によって1つのトークンにマッピングされた。<M>は共有マスクトークン、<X>、<Y>、<Z>は固有のトークンIDが割り当てられたセンチネルトークンであることを示す。BERTスタイルの目的（2行目）には、一部のトークンがランダムなトークンIDで置き換えられるという問題が含まれています（グレーアウトしたリンゴという単語でこれを示しています）。

目的	グリュー	シーエヌエヌエム	スクワッド	スグレもの	エンデ	エヌファール	エンロ
接頭辞言語モデリング	80.69	18.94	77.99	65.27	**26.86**	39.73	**27.49**
BERTスタイル（Devlinら、2018年）	**82.96**	**19.17**	**80.65**	**69.85**	26.78	**40.03**	27.41
デスハフリング	73.17	18.59	67.61	58.47	26.11	39.30	25.62

表4：3.3.1節で説明した3つの異質な事前学習目的のパフォーマンス。

をデコーダーで予測する。第二に、BERT（Devlin et al., 2018）で使用されている「masked language modeling」（MLM）目的に触発された目的を検討する。MLMは、テキストのスパンを取り、トークンの15%を破損させる。破損したト

ークンの90%は特別なマスクトークンに置き換えられ、10%はランダムなトークンに置き換わります。BERTはエンコーダのみのモデルであるため、事前学習時の目標は、エンコーダの出力でマスクされたトークンを再構築することである。エンコーダ・デコーダの場合は、単純に破損していないシーケンス全体をターゲットとして使用する。これは、破損したトークンのみをターゲットとするベースラインの目的とは異なることに注意されたい。この2つのアプローチをセクション3.3.2で比較する。最後に、我々は、例えば(Liu et al., 2019a)で使用され、それがノイズ除去シーケンシャルオートエンコーダーに適用されたような基本的なデシャフリング目的も検討する。このアプローチは、トークンのシーケンスを取り、それをシャッフルし、そしてターゲットとして元のdeshuffledシーケンスを使用する。表3の最初の3行に、これら3つの手法の入力とターゲットの例を示す。

これら3つの目的の性能を表4に示す。全体として、BERTスタイルの目的が最も良い性能を発揮していることがわかりますが、翻訳タスクでは、前置言語モデリング目的が同様の性能を達成しています。実際、BERT目的の動機は、言語モデルベースの事前学習を凌駕することであった。deshuffling目的は、prefix language modelingとBERT-style目的の両方よりもかなり悪い結果となっています。

目的	グリュー	シーエヌエヌエム	スクワッド	スグレもの	EnDe	EnFr	エンロ
BERT式（デブリンら、2018年）	82.96	19.17	**80.65**	69.85	26.78	**40.03**	27.41
MASSスタイル（Song et al, 2019)	82.32	19.16	80.10	69.28	26.79	**39.89**	27.55
F 破損したスパンを置き換える	83.28	**19.24**	**80.88**	**71.36**	**26.98**	39.82	**27.65**
破損したトークンを落とす	**84.44**	**19.31**	**80.52**	68.67	**27.07**	39.76	**27.82**

表5：BERTスタイルの事前学習目的のバリエーション比較。最初の2つのバリエーションでは、モデルは破損していない元のテキストセグメントを再構築するように訓練される。後者の2つでは、モデルは破損したトークンのシーケンスのみを予測する。

3.3.2. BERTの目的の簡略化

前節の結果に基づき、ここでは、BERTスタイルのノイズ除去目的の修正について検討することに焦点を当てます。この目的は、もともと、分類とスパン予測のために訓練されたエンコーダのみのモデルのための事前訓練技術として提案されました。そのため、エンコーダとデコーダのテキストツーテキストセットアップにおいて、より良いパフォーマンスやより効率的になるように修正することが可能かもしれません。

まず、BERTスタイルの目的語の単純な変形を検討し、ランダムなトークン交換のステップを含めないようにします。結果として得られる目的は、単に入力のトークンの15%をマスクトークンに置き換え、モデルは、破壊されていない元のシーケンスを再構築するように訓練される。同様のマスク目的語がSongら（2019）によって使用され、そこで「MASS」と呼ばれたので、この変種を「MASS-style」目的語と呼ぶことにする。第二に、我々は、これがデコーダにおける長いシーケンスにわたって自己注意を必要とするので、破損していないテキストスパン全体を予測することを避けることが可能であるかどうかを確認することに興味があった。これを実現するために、2つの戦略を検討した：まず、各破損トークンをマスクトークンに置

き換える代わりに、各破損トークンの連続するスパン全体をユニークなマスクトークンに置き換えます。そして、ターゲットとなるシーケンスは、「破損した」スパンを連結したものとなり、それぞれのスパンには、入力の置換に使われたマスクトークンが前置される。これは、セクション3.1.4で説明するベースラインで使用する事前トレーニングの目的である。次に、入力シーケンスから破損したトークンを完全に削除し、削除されたトークンを順番に再構築することをモデルに課すという方法も検討する。これらのアプローチの例は、表3の5行目と6行目に示されている。

オリジナルの BERT スタイルの目的とこれら 3 つの代替案との経験的な比較を表 5 に示します。私たちの設定において、これらの代替案はすべて同様の性能を発揮することがわかります。唯一の例外は、破損したトークンを完全に削除すると、CoLA のスコアが大幅に向上するため、GLUE スコアがわずかに向上することです（ベースライン平均の 53.84 に対し 60.04 、表 16 参照）。これは、CoLAが文法的・構文的に許容される文章かどうかを分類するものであり、トークンの欠落を判断できることが許容度の検出と密接に関係しているためと思われます。しかし、SuperGLUEでは、トークンを完全に削除すると、センチネルトークンに置き換えるよりも成績が悪くなった。元の配列を完全に予測する必要がない2つの変形（"replace corrupted spans" と "drop corrupted spans"）は、ターゲット配列を短くすることができ、その結果、トレーニングが容易になるため、潜在的に魅力的である。

	汚職率	グリュー	シーエヌエヌエム	スクワッド	スグレもの	エンデ	エヌフアール	エンロ
	10%	**82.82**	19.00	**80.38**	69.55	**26.87**	39.28	**27.44**
F	15%	**83.28**	19.24	**80.88**	**71.36**	**26.98**	**39.82**	**27.65**
	25%	**83.00**	**19.54**	**80.96**	70.48	**27.04**	**39.83**	**27.47**
	50%	81.27	19.32	79.80	70.33	**27.01**	**39.90**	**27.49**

表 6: i.i.d. corruption objective の異なる汚職率でのパフォーマンス。

を高速化した。今後は、破損したスパンをセンチネルトークンに置き換え、破損したトークンのみを予測する（ベースラインの目的と同じ）バリエーションを検討する予定です。

3.3.3. 腐敗率を変化させる

これまで、私たちはトークンの15%を破損してきましたが、これはBERT（Devlin et al., 2018）で使用されている値です。ここでも、私たちのtext-to-textフレームワークはBERTのものとは異なるため、異なる破損率が私たちにとってより良く機能するかどうかを確認することに興味があります。表6で、10%、15%、25%、50%の破損率を比較します。全体として、破損率がモデルの性能に与える影響は限定的であることがわかります。唯一の例外は、我々が検討した最大の破損率（50%）で、GLUEとSQuADの性能が大幅に低下したことです。また、より大きな破損率を使用すると、ターゲットが長くなり、トレーニングが遅くなる可能性があります。これらの結果とBERTが設定した歴史的な前例に基づき、今後、破損率15%を使用することにします。

3.3.4. スパンを腐らせる

次に、より短いターゲットを予測することで、トレーニングのスピードを上げるという目標に目を向けます。これまでのアプローチでは、各入力トークンを破損させるかどうかをi.i.d.で判断している。複数の連続したトークンが破損した場合、それらは「スパン」として扱われ、単一のユニークなマスクトークンがスパン全体を置

き換えるために使用されます。スパン全体を単一のトークンで置き換えることで、ラベルのないテキストデータをより短いシーケンスに処理することができます。i.i.d.破損戦略を使用しているため、破損したトークンが連続して現れるとは限りません。そのため、個々のトークンをi.i.d.方式で破損させるのではなく、トークンのスパンを特別に破損させることで、さらなる高速化を実現できる可能性があります。スパンを破損することは、以前、BERTの事前訓練目的としても検討され、パフォーマンスを向上させることが判明した（Joshi et al.、2019）。

このアイデアを検証するために、連続したランダムな間隔のトークンのスパンを特に破損させる目的を考えてみる。この目的は、破損させるトークンの割合と破損させるスパンの総数でパラメタ化できる。そして、これらの指定されたパラメータを満たすように、スパンの長さがランダムに選択される。例えば、500個のトークンのシーケンスを処理する場合、トークンの15%を破損させ、かつ
が25スパンであれば、破損したトークンの総数は500×0.15＝75、平均スパン長は75/25＝3である。なお、元の配列が与えられた場合
長さと破損率から、この目的を等価的に平均スパン長または総スパン数でパラメトリック化することができます。

スパン長	グリュー	シーエヌエヌエム	スクワッド	スグレもの	エンデ	エヌフアール	エンロ
F ベースライン（i.i.d.）	**83.28**	19.24	80.88	71.36	**26.98**	**39.82**	**27.65**
2	**83.54**	19.39	**82.09**	**72.20**	**26.76**	**39.99**	**27.63**
3	**83.49**	**19.62**	**81.84**	**72.53**	**26.86**	39.65	**27.62**
5	**83.40**	19.24	**82.05**	**72.23**	**26.88**	39.40	**27.53**
10	82.85	19.33	**81.84**	70.44	**26.79**	39.49	**27.69**

表7：異なる平均スパン長に対するスパン破損目的（Joshi et al.（2019）に触発された）の性能。すべてのケースで、元のテキストシーケンスの15%を破損させる。

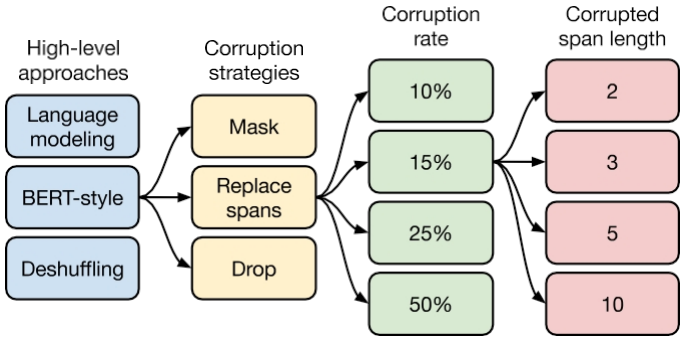

図5：教師なし目的の探求のフローチャート。まず、セクション3.3.1において、いくつかの異なるアプローチを検討し、BERTスタイルのノイズ除去目的が最もよく機能することを発見した。次に、セクション3.3.2において、より短いターゲットシーケンスを生成するためにBERT目的を簡略化するための様々な方法を検討する。脱落したスパンをセンチネルトークンで置き換えることで、短いターゲット配列が得られることを踏まえ、セクション 3.3.3 では、異なる破損率について実験する。最後に、セクション3.3.4で、連続したトークンのスパンを意図的に破損させる目的語を評価する。

表7では、スパンコラプションとi.i.dコラプションを比較しています。すべて

のケースで破損率15%を使用し、平均スパン長を2、3、5、10として比較しました。しかし、平均スパン長を10とした場合、他の値より若干劣る場合がある。また、特に平均スパン長を3にすると、翻訳以外のほとんどのベンチマークでi.i.d.目的よりわずかに（しかし有意に）優れていることがわかります。幸いなことに、スパン破損は平均的に短い配列を生成するため、i.i.d.ノイズアプローチと比較して学習時のスピードアップも実現する。

3.3.5. ディスカッション

図5は、教師なし目的の探索で行った選択のフローチャートである。全体として、私たちが観察した性能の最も大きな違いは、次のとおりです。

は、事前学習において言語モデリングとデシャッフルを上回った。我々は、探索したノイズ除去目的の多くのバリエーションにおいて、顕著な差は観察されなかった。しかし、異なる目的（または目的のパラメータ化）は、異なるシーケンス長をもたらし、したがって、異なるトレーニング速度をもたらす可能性があります。このことは、今回検討したノイズ除去目的の選択は、主に計算コストに応じて行う必要があることを示唆しています。また、我々の結果は、今回検討したものと同様の目的をさらに検討しても、我々が検討したタスクやモデルにおいて大きな利益にはつながらない可能性があることを示唆している。むしろ、ラベルなしデータを活用する全く別の方法を探ることが僥倖である可能性がある。

3.4. 事前学習 データセット

教師なし学習と同様に、事前学習データセット自体も、転移学習パイプラインの重要な構成要素です。しかし、目的やベンチマークとは異なり、新しい事前学習用データセットは、それ自体が重要な貢献として扱われることはなく、事前学習済みのモデルやコードと一緒に公開されないことがよくあります。その代わりに、新しい手法やモデルを紹介する際に導入されるのが一般的です。そのため、異なる事前学習用データセットの比較は比較的少なく、事前学習に使用される「標準的な」データセットも存在しません。最近の注目すべき例外（Baevski et al., 2019; Liu et al., 2019c; Yang et al., 2019）は、新しい大規模（しばしばCommon Crawlソース）データセットでの事前学習と、より小さな既存のデータセット（しばしばWikipedia）を使用した事前学習を比較しています。事前学習データセットが性能に与える影響をより深く探るために、このセクションでは、我々のC4データセットと他の潜在的な事前学習データ源の変種を比較する。私たちは、TensorFlow Datasets.[11]の一部として、検討したC4データセットのバリエーションをすべて公開しています。

3.4.1. 非標識データセット

C4の作成にあたり、Common Crawlから抽出されたWebテキストをフィルタリングするための様々なヒューリスティックを開発した（説明についてはセクション2.2を参照

）。我々は、他のフィルタリングアプローチや一般的な事前学習データセットとの比較に加え、このフィルタリングが下流タスクの性能向上につながるかどうかを測定することに興味がある。この目的のために、以下のデータセットで事前学習後のベースラインモデルの性能を比較する：

C4 ベースラインとして、まず、セクション2.2で説明したように、提案したラベルなしデータセットでの事前学習を検討します。

フィルタリングされていないC4 C4を作成する際に用いたヒューリスティックなフィルタリング（重複排除、悪い単語の除去、文のみの保持など）の効果を測定するために、このフィルタリングを省いたC4の代替バージョンも作成しました。ただし、英文の抽出にはlangdetectを使用しています。その結果、langdetectは非自然的な英文テキストに低い確率を割り当てることがあるため、「フィルタリングなし」バージョンにもフィルタリングが含まれています。

RealNews-like 最近の研究では、ニュースウェブサイトから抽出されたテキストデータを使用している（Zellers et al.）このアプローチと比較するために、「RealNews」データセット（Zellers et al., 2019）で使用されたドメインの1つからのコンテンツのみを含むようにC4を追加的にフィルタリングして、別のラベルなしデータセットを生成します。なお、簡単にするために

11. https://www.tensorflow.org/datasets/catalog/c4

	データセット	サイズ	グリュー	シーエヌエヌエム	スクワッド	スグレもの	エンデ	エヌフアール	エンロ
F	C4	745GB	83.28	**19.24**	80.88	71.36	**26.98**	**39.82**	**27.65**
	C4、アンフィルタード	6.1TB	81.46	19.14	78.78	68.04	26.55	39.34	27.21
	リアルニュースのような	35GB	**83.83**	**19.23**	80.39	72.38	**26.75**	**39.90**	**27.48**
	ウェブテキスト的	17GB	**84.03**	**19.31**	**81.42**	71.40	**26.80**	**39.74**	**27.59**
	ウィキペディア	16GB	81.85	**19.31**	81.29	68.01	**26.94**	39.69	**27.67**
	ウィキペディア + TBC	20GB	83.65	**19.28**	**82.08**	**73.24**	**26.77**	39.63	**27.57**

表8：異なるデータセットで事前学習を行った結果の性能。最初の4つのバリエーションは、新しいC4データセットに基づくものである。

の比較では、C4で使用されたヒューリスティック・フィルタリングの手法を踏襲しています。唯一の違いは、表向きはニュース以外のコンテンツを排除していることです。

WebText-like 同様に、WebTextデータセット（Radford et al., 2019）は、コンテンツ集約サイトRedditに投稿され、少なくとも3の「スコア」を得たウェブページのコンテンツのみを使用しています。 Redditに投稿されたウェブページのスコアは、ウェブページを支持（アップボート）または反対（ダウンボート）したユーザーの割合に基づいて計算されています。Redditのスコアを品質シグナルとして使用する背景には、同サイトのユーザーは高品質のテキストコンテンツにのみアップヴォートするという考えがあります。比較可能なデータセットを作成するために、まず、C4から、OpenWebTextが作成したリストに掲載されているURLから発信されていないコンテンツをすべて削除してみた[12]。しかし、ほとんどのページがRedditに登場しないため、結果として約2GBと、比較的少ないコンテンツになった。C4は、Common Crawlの1ヶ月分のデータに基づいて作成されたことを思い出してください。そこで、法外に小さなデータセットを使用することを避けるため、Common Crawlから2018年8月から2019年7月までの12か月分のデータをダウンロードし、C4用のヒューリスティック・フィルタリ

ングを適用し、Redditフィルタを適用しました。これにより、17GBのWebTextのようなデータセットが生成され、これはオリジナルの40GBのWebTextデータセット（Radford et al.、2019）と同等のサイズとなりました。

ウィキペディア ウェブサイト「ウィキペディア」は、共同で書かれた何百万もの百科事典の記事で構成されています。このサイトのコンテンツは厳格な品質ガイドラインの対象であるため、クリーンで自然なテキストの信頼できるソースとして使用されている。我々は、TensorFlow Datasets[13]の英語版Wikipediaのテキストデータを使用し、記事からマークアップや参照セクションを省略する。

Wikipedia + Toronto Books Corpus Wikipediaの事前学習データを使用することの欠点は、自然テキストの1つの可能なドメイン（百科事典の記事）しか表していないことです。

これを軽減するために、BERT（Devlin et al., 2018）は、Wikipediaからのデータを Toronto Books Corpus（TBC）（Zhuら、2015）と組み合わせました。TBCは電子書籍から抽出されたテキストを含んでおり、自然言語の異なるドメインを表しています。BERTの人気により、Wikipedia + TBCの組み合わせは、その後の多くの作品で使用されるようになりました。

12. https://github.com/jcpeterson/openwebtext
13. https://www.tensorflow.org/datasets/catalog/wikipedia

これらの各データセットで事前学習を行った結果を表8に示します。まず明らかなのは、C4からヒューリスティック・フィルタリングを取り除くと、性能が一様に低下し、フィルタリングなしのバリアントがすべてのタスクで最悪の性能を示すということです。これ以外にも、より制約の多いドメインを持つ事前学習用データセットが、多様なC4データセットを凌駕するケースもあることがわかりました。例えば、Wikipedia + TBCコーパスを使用した場合、SuperGLUEのスコアは73.24となり、ベースラインのスコア（C4使用）71.36を上回りました。これは、ほぼ完全に、25.78（ベースライン、C4）のパフォーマンスが向上したことに起因しています。

MultiRCのExact Matchスコアでは50.93（Wikipedia + TBC）であった（表16参照）。MultiRCは読解のデータセットであり、その最大のデータ源は小説本であり、まさにTBCのカバーする領域である。同様に、ニュース記事の読解力を測定するデータセットであるReCoRDの事前学習において、RealNewsのようなデータセットを使用することで、完全一致スコアが68.16から73.72に増加した。最後の例として、Wikipediaのデータを使用することで、Wikipediaの文章を使用した質問応答データセットであるSQuADにおいて、有意な（しかしそれほど劇的ではない）向上が見られました。同様の観察は先行研究でも行われており、例えばBeltagyら（2019）は、研究論文からのテキストでBERTを事前訓練すると、科学的タスクでのパフォーマンスが向上することを発見した。これらの知見の背後にある主な教訓は、*ドメイン内のラベルなしデータに対する事前訓練は、下流タスクのパフォーマンスを向上させることができるということです*。これは驚くべきことではないが、我々の目標が任意のドメインからの言語タスクに迅速に適応できるモデルを事前訓練することであるならば、不満足でもある。Liuら（2019c）は、より多様なデータセットで事前学習することで、ダウンストリームタスクでの改善がもたらされることも観察した。この観察は、自然言語処理のドメイン適応に関する並行研究の動機にもなっている。

例：Ruder (2019); Li (2012).

単一ドメインでの事前学習だけでは、結果として得られるデータセットが大

幅に小さくなることが多いという欠点があります。同様に、WebTextのようなバリアントも同じかそれ以上の性能を発揮しました。

Redditベースのフィルタリングは、Common Crawlの12倍以上のデータに基づいているにもかかわらず、C4のデータセットよりも約40倍小さいデータセットを生成しています。ただし、ベースラインの設定では、2^{35}≈34B□のデータセットに対してのみ事前学習を行っています。
トークンは、我々が考える最小の事前学習データセットの約8倍しかない。
どのような場合に、より少ない事前学習データセットを使用することが問題になるかを、次のセクションで調査します。

3.4.2. 事前学習 データセットサイズ

C4の作成に使用しているパイプラインは、非常に大きな事前学習データセットを作成できるように設計されています。このように多くのデータを利用することで、モデルを繰り返し学習させることなく、事前学習を行うことができます。事前トレーニングの目的は確率的なものであり、モデルが同じデータを何度も見ることを防ぐことができるため、事前トレーニング中に例を繰り返すことが、下流の性能に役立つのか有害なのかは明らかではありません。

限られたラベルなしデータセットサイズの効果を検証するために、ベースラインモデルをC4の人為的な切り捨てバージョンで事前トレーニングしました。ベースラインモデルを2^{35}≈34Bトークン（C4全体のサイズのごく一部）で事前訓練したことを思い出してください。切り詰めたトークンでのトレーニングは2^{29}、2^{27}、2^{25}、2^{23}のトークンからなるC4のバリアント。これらのサイズは、繰り返しを64回、256回、1,024回、4,096回、それぞれ学習させました。

トークンの数	リピート数	グルー	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
フルデータセット	0	**83.28**	**19.24**	**80.88**	**71.36**	**26.98**	**39.82**	**27.65**
229	64	**82.87**	**19.19**	**80.97**	**72.03**	**26.83**	**39.74**	**27.63**
227	256	82.62	**19.20**	79.78	69.97	**27.02**	**39.71**	27.33
225	1,024	79.55	18.57	76.27	64.76	26.38	39.56	26.80
223	4,096	76.34	18.33	70.92	59.29	26.37	38.84	25.81

表9：事前学習時のデータの繰り返しの効果測定。これらの実験では、C4から最初の*N個の*トークンのみを使用していますが（*Nの値は*1列目に示されています）、それでも2^{35}個のトークンに対してプリトレーニングを行います。この結果、事前学習の過程でデータセットが繰り返されることになり（各実験の繰り返し回数は2列目に示されている）、暗記につながる可能性がある（図6参照）。

その結果、ダウンストリーム性能は表9に示すようになった。予想通り、データセットサイズが小さくなるにつれて性能が低下している。これは、モデルがトレーニング前のデータセットを記憶し始めたためではないかと推測されます。このことが本当かどうかを調べるために、図6にデータセットサイズごとの学習損失をプロットしてみました。実際、学習前のデータセットのサイズが小さくなるにつれて、モデルは著しく小さな学習損失を達成し、記憶化の可能性を示唆している。Baevskiら(2019)も同様に、事前学習データセットサイズを切り捨てることで、下流のタスクパフォーマンスを低下させることを観察している。

なお、これらの効果は、事前学習用データセットを64回だけ繰り返した場合に限定的である。このことは、事前学習データのある程度の繰り返しが有害でないことを示唆している。しかし、追加の事前学習は有益であり（セクション3.6で示す）、追加のラベルなしデータの入手は安価で簡単であることから、可能な限り大きな事前学習データセットを使用することを提案する。また、この効果はモデルサイズが大きいほど顕著である。つまり、大きなモデルは小さな事前学習データセットに対してよりオーバーフィットしやすくなる可能性がある。

3.5. トレーニング戦略

これまで、教師なしタスクでモデルの全パラメータを事前学習した後、教師ありタスクで微調整を行うという設定を考えてきた。このアプローチは簡単であるが、ダウンストリーム/教師ありタスクでモデルを訓練するための様々な代替方法が提案されている。本節では、複数のタスクで同時にモデルを訓練するアプローチに加え、モデルを微調整するための様々なスキームを比較する。

3.5.1. ファインチューニング方法

モデルのパラメータをすべて微調整することは、特に低リソースタスクにおいて、最適でない結果をもたらす可能性があることが議論されている（Peters et al.、2019）。テキスト分類タスクのための転移学習に関する初期の結果は、固定された事前学習済みモデルによって生成された文埋め込みを与えられた小さな分類器のパラメータのみを微調整することを提唱した（Subra- manian et al., 2018; Kiros et al., 2015; Logeswaran and Lee, 2018; Hill et al.

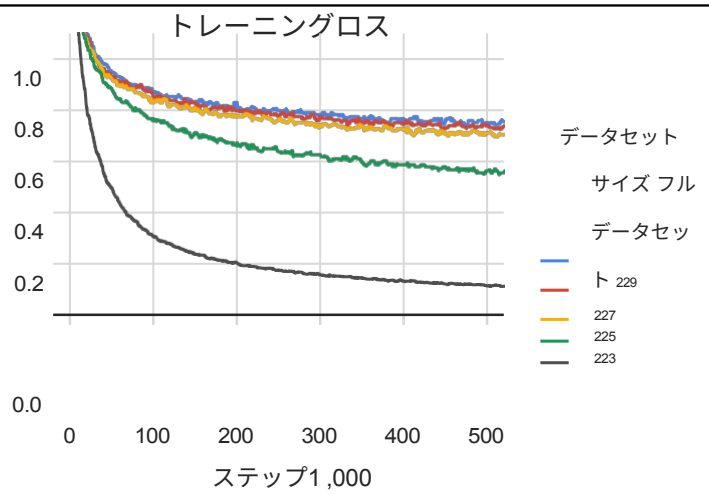

図6：オリジナルのC4データセットと、4つの人工的な切り捨てを行ったデータセットの学習前損失。記載されているサイズは、各データセットのトークンの数を表しています。4つのサイズは、事前学習の過程でデータセットを64回から4,096回繰り返すことに対応しています。データセットのサイズを小さくすると、トレーニングの損失値が小さくなり、ラベルのないデータセットを記憶していることが示唆されます。

ら、2017年）。このアプローチは、与えられたタスクのターゲットシーケンスを出力するためにデコーダ全体を訓練する必要があるため、我々のエンコーダデコーダモデルにはあまり適用できません。その代わりに、エンコーダデコーダモデルのパラメータのサブセットのみを更新する、2つの代替的な微調整アプローチに焦点を当てます。

1つ目の「アダプター層」（Houlsby et al, 2019; Bapna et al, 2019）は、微調整をしながら元のモデルの大部分を固定しておくという目標が動機となっています。アダプターレイヤーは、Transformerの各ブロックの既存のフィードフォワードネットワークの後に追加される、密な-ReLU-密なブロックである。これらの新しいフィードフォワードネットワークは、その出力の次元が入力と一致するように設計されています。このため、構造やパラメータを変更することなく、ネッ

トワークに挿入することができます。微調整の際には、アダプター層と層の正規化パラメータのみが更新されます。このアプローチの主なハイパーパラメータは、フィードフォワードネットワークの内部次元数dであり、モデルに追加される新しいパラメータの数を変更する。dの値をいろいろ変えて実験してみた。

我々が考える2つ目の代替微調整法は、「gradual unfreezing」（Howard and Ruder, 2018）である。gradual unfreezingでは、時間の経過とともに、より多くのモデルのパラメータが微調整される。Gradual unfreezingはもともと、1つのスタック層からなる言語モデルアーキテクチャに適用されていた。この設定では、微調整の開始時に最終層のパラメータのみを更新し、一定回数の更新を行った後に最後から2番目の層のパラメータも更新し、ネットワーク全体のパラメータが微調整されるまで、この作業を繰り返します。この方法をエンコーダ・デコーダのモデルに適用するため、エンコーダとデコーダの層の凍結を、並行して上から順に徐々に解除していきます。入力埋め込み行列と出力分類行列のパラメータは共有されているため、微調整の間、パラメータを更新します。ベースラインモデルはエンコーダとデコーダそれぞれ12層で構成され、次のような微調整が行われていることを思い出してください。

ファインチューニング方式		グルー	CNNDM	SQuAD	SGLUE	EnDe	EnFr	EnRo
F	全パラメーター	**83.28**	**19.24**	**80.88**	**71.36**	**26.98**	**39.82**	**27.65**
	アダプター層、d = 32	80.52	15.08	79.32	60.40	13.84	17.88	15.54
	アダプター層、d = 128	81.51	16.62	79.47	63.03	19.83	27.50	22.63
	アダプター層、d = 512	81.54	17.78	79.18	64.30	23.45	33.98	25.81
	アダプター層、d = 2048	81.51	16.62	79.47	63.03	19.83	27.50	22.63
	徐々に凍結を解除する	82.50	18.95	79.17	**70.79**	26.71	39.02	26.93

表10：モデルのパラメータのサブセットのみを更新する、さまざまな代替微調整方法の比較。アダプター層については、*d*はアダプターの内部次元を意味する。

^{218}ステップである。そのため、微調整プロセスを$^{218}/_{12}$ステップずつの12エピソードに細分化し、*n番目の*エピソードで12層-nから12層まで訓練します。Howard and Ruder (2018)は、トレーニングの各エポック後に追加のレイヤーをファインチューニングすることを提案していることに留意します。ただし、以下のように

また、GLUEやSuperGLUEのように、多くのタスクが混在しているタスクもあるため、$^{218}/_{12}$ステップごとにレイヤーを追加して微調整するというシンプルな戦略を採用しています。

これらの微調整手法の性能比較を表10に示す。アダプター層については、32、128、512、2048の内部次元*d*を使用してパフォーマンスを報告する。過去の結果（Houlsby et al., 2019; Bapna et al., 2019）に従い、SQuADのような低リソースタスクは小さな値の*d*でうまく機能するのに対し、高リソースタスクは妥当なパフォーマンスを得るために大きな次元を必要とすることが分かりました。これは、次元性がタスクサイズに適切にスケーリングされる限り、アダプターレイヤーがより少ないパラメータで微調整するための有望な手法になり得ることを示唆しています。GLUE と SuperGLUE は、それぞれのデータセットを連結して一つの「タスク」として扱っているため、低リソースデータセットで構成されていますが、連結したデータセットは十分大きいため、大きな *d* 値が必要です。凍結解除のスケジュールをより注意深く調整することで、より良い結果が得られるかもしれません。

3.5.2. マルチタスク学習

これまでは、単一の教師なし学習タスクでモデルを事前学習した後に、下流の各タスクで個別に微調整していました。別のアプローチとして、「マルチタスク学習」（Ruder, 2017; Caruana, 1997）と呼ばれる、一度に複数のタスクでモデルを訓練することがあります。このアプローチは通常、同時に多くのタスクを実行できる単一のモデルを同時に訓練する、すなわち、モデルとそのパラメータの大部分をすべてのタスクで共有することを目標としている。私たちはこの目標をやや緩和し、その代わりに複数のタスクを同時にトレーニングする方法を調査し、最終的に個々のタスクでうまく機能する別々のパラメータ設定を作成することを目的としています。例えば、1つのモデルを多くのタスクで学習させ、性能を報告する際には、タスクごとに異なるチェックポイントを選択することができるようにすることが考えられます。このように、マルチタスク学習の枠組みを緩めることで、これまで検討してきた「事前学習→微調整」のアプローチと比較して、より公平な立場に立つことができます。また、私たちの

text-to-textの統一的な枠組みでは、「マルチタスク学習」は、単にデータセットを混ぜ合わせることに相当する。つまり、教師なしタスクを混合されたタスクの1つとして扱うことで、マルチタスク学習でもラベルのないデータに対して学習することができるのである。これに対して、マルチタスク学習のNLPへの応用の多くは、タスク固有の分類ネットワークを追加したり、タスクごとに異なる損失関数を使用したりします（Liu et al., 2019b）。

Arivazhaganら（2019）が指摘するように、マルチタスク学習において非常に重要な要素は、各タスクのデータをどれだけモデルに学習させるかです。つまり、モデルが与えられたタスクのデータを十分に見て、そのタスクをうまくこなせるようにしたいが、トレーニングセットを記憶してしまうほど多くのデータを見ることは避けたい。各タスクのデータの割合をどのように設定するかは、データセットのサイズ、タスクの学習の「難易度」（すなわち、モデルがタスクを効果的に実行できるようになるまでにどれだけのデータを見る必要があるか）、正則化など、さまざまな要因によって決まる。さらに、「タスク干渉」や「ネガティブトランスファー」と呼ばれる、あるタスクで優れた性能を発揮すると、別のタスクの性能を阻害する可能性があることも問題です。このような懸念があるため、まず、各タスクから得られるデータの割合を設定するための様々な戦略を検討する。同様の探求は、Wangら（2019a）によって行われました。

例-比例混合 あるタスクに対してモデルがどれだけ早くオーバーフィットするかは、そのタスクのデータセットサイズに大きく影響されます。そのため、混合比率を設定する自然な方法は、各タスクのデータセットのサイズに比例してサンプリングすることです。これは、全タスクのデータセットを連結し、連結されたデータセットからランダムに例を抽出することと同じである。ただし、教師なしノイズ除去タスクは、他のタスクのデータセットよりも桁違いに大きいデータセットを使用しています。つまり、各データセットのサイズに比例してサンプリングすると、モデルが見るデータの大部分はラベルなしとなり、すべての教師ありタスクで学習不足となるのです。教師

なしタスクがない場合でも、一部のタスク（例えばWMT英仏）は非常に大規模であるため、同様にほとんどのバッチをクラウド化してしまう。この問題を回避するために、比率を計算する前にデータセットサイズに人為的な「制限」を設けます。具体的には、それぞれのデータセットに含まれる例の数が

N個のタスクのデータセットをe_n、$n \in \{1, ...N\}$とすると、学習中にm番目のタスクから例をサンプリングする確率を$r_m = \min(e_m, K) / \sum \min(e_n, K)$とするとき。

Kは人工データセットのサイズ制限です。

温度スケーリング混合 データセットサイズ間の大きな格差を緩和する別の方法は、混合率の「温度」を調整することである。このアプローチは、多言語BERT で、低リソース言語でモデルが十分に訓練されていることを確認するために使用されました[14]。温度スケーリングを温度 T で実装するには、以下のようにします。

各タスクの混合率r_mを$1/T$のべき乗にし、合計が1になるように再正規化する。$T = 1$のとき、この方法は例題比例混合と同じである。

であり、Tが大きくなるにつれて、その比率はより等しい混合に近づいていく。データセットサイズの制限K（温度スケーリング前にr_mを得るために適用）はそのままに、$K = 2^{21}$という大きな値に設定しました。温度を上げると最大のデータセットの混合率が低下するため、Kの値を大きくしています。

14. https://github.com/google-research/bert/blob/master/multilingual.md

ミキシング戦略	グリュー	シーエヌエヌ	スクワッド	SGLUE	EnDe	EnFr	エンロ
イコール	76.13	19.02	76.51	63.37	23.89	34.31	26.78
例-比例型、$K = 2^{16}$	80.45	19.01	77.25	69.95	24.35	34.99	27.10
例-比例型、$K = 2^{17}$	**81.28**	**19.24**	**80.88**	**71.36**	**24.98**	**35.82**	**27.65**
例-比例型、$K = 2^{18}$	81.50	19.12	77.00	67.94	24.30	35.00	27.25
例-比例型、$K = 2^{19}$	81.42	**19.24**	79.78	67.30	25.21	36.30	**27.76**
例-比例型、$K = 2^{20}$	80.80	**19.24**	**80.36**	67.38	25.66	36.93	**27.68**
例-比例型、$K = 2^{21}$	79.83	18.79	79.50	65.10	25.82	37.22	27.13
温度スケール、$T = 2$	81.90	**19.28**	79.42	69.92	25.42	36.72	27.20
温度スケーリング、$T = 4$	80.56	**19.22**	77.99	69.54	25.04	35.82	27.45
温度スケーリング、$T = 8$	77.21	19.10	77.14	66.07	24.55	35.35	27.17

表11：異なるミキシング戦略を用いたマルチタスクトレーニングの比較。例-比例混合とは、各データセットの合計サイズに応じて各データセットから例をサンプリングすることであり、最大データセットサイズには人為的制限（K）がある。*温度*スケールミキシングは、サンプリングレートを温度Tで再スケーリングするものである。温度差混合では、人工的なデータセットサイズの制限である$K = 2^{21}$を使用します。

均等混合 この場合、各タスクから等確率でサンプルを採取する。具体的には、各バッチの各例は、訓練するデータセットの1つから一様にランダムにサンプリングされます。これは、低リソースタスクではモデルがすぐにオーバーフィットし、高リソースタスクではアンダーフィットするため、最適とは言えない戦略である可能性が高い。ここでは、比率を最適でない方法で設定した場合に、どのような問題が生じるかを示す参考資料として、この方法を紹介します。

これらのミキシング戦略を、ベースラインである「訓練前-訓練後-微調整」の結果と同等に比較するために、同じ総ステップ数でマルチタスクモデルを訓練します：$2^{19} + 2^{18} = 786{,}432$.その結果を表11に示す。

一般に、マルチタスクトレーニングは、ほとんどのタスクにおいて、事前トレーニングの後に微調整を行うよりも劣っていることがわかります。これは、低リソースタスクがオーバーフィットした、高リソースタスクが十分なデータを見ていない、あるいは、モデルが汎用的な言語能力を学習するのに十分なラベルなしデータ

を見ていないことが原因である可能性があります。例-比例混合では、ほとんどのタスクで、モデルが最高のパフォーマンスを得るKの「スイートスポット」が存在し、Kの値が大きくても小さくても、パフォーマンスが低下する傾向があることがわかった。例外は、WMT英仏翻訳で、これは高リソースタスクであるため、混合比率を高くすることで常に利益を得ることができます。最後に、温度スケールの混合は、ほとんどのタスクで妥当な性能を得るための手段であり、ほとんどのケースで$T = 2$が最も良い性能を発揮することに留意されたい。マルチタスクモデルが、個々のタスクで訓練された個別のモデルよりも優れているという発見は、例えば、Arivazhaganら（2019）およびMcCannら（2018）によって以前に観察されているが、マルチタスク設定は、非常に類似したタスクにわたって利益をもたらすことが示されているLiuら（2019b）; Ratnerら（2018）であった。以下のセクションでは、マルチタスクトレーニングとプレトレーニング・テンファインチューニングアプローチの間のギャップを埋める方法を探ります。

トレーニング戦略	グリュー	シーエヌエヌエム	スクワッド	スグレもの	エンデ	エヌファール	エンロ
F 教師なし事前学習 + ファインチューニング	**83.28**	**19.24**	**80.88**	**71.36**	**26.98**	39.82	27.65
マルチタスクトレーニング	81.42	**19.24**	79.78	67.30	25.21	36.30	27.76
マルチタスクの事前トレーニング + ファインチューニング	**83.11**	**19.12**	**80.26**	**71.03**	**27.08**	39.80	**28.07**
放置型マルチタスクトレーニング	81.98	19.05	79.97	**71.68**	**26.93**	39.79	**27.87**
教師ありのマルチタスク事前学習	79.93	18.96	77.38	65.36	26.81	**40.13**	**28.04**

表12 教師なし事前学習、マルチタスク学習、様々な形態のマルチタスク事前学習の比較。

3.5.3. マルチタスク学習とファインチューニングの組み合わせ

ここでは、マルチタスク学習の緩和版として、タスクの混合に対して単一のモデルを訓練するが、モデルの異なるパラメータ設定（チェックポイント）を用いて性能を評価することを認めていることを思い出してほしい。このアプローチを拡張して、一度にすべてのタスクに対してモデルを事前学習させ、その後、個々のタスクに対して微調整を行うケースを考えることができる。これは、導入時にGLUEや他のベンチマークで最先端の性能を達成した「MT-DNN」（Liu et al.、2015、2019b）が採用している方法である。この手法の3つのバリエーションを検討する：1つ目は、個々の下流タスクで微調整する前に、$K = 2^{19}$の人工データセットサイズ制限を持つ例-比例混合でモデルを単純に事前訓練します。これにより、教師ありタスクと教師なしタスクを事前訓練に加えることで、モデルが下流タスクに早期に触れることができるかどうかを測定することができます。また、多くの監督元を混ぜることで、事前学習したモデルが個々のタスクに適応する前に、より一般的な「スキル」（大雑把に言えば）を獲得することができると期待できるかもしれません。これを直接測定するために、同じ例-比例混合（$K = 2^{19}$）でモデルを事前訓練するが、この事前訓練混合から下流のタスクを1つ省くという第2のバリエーションを考えてみることにする。次に、プレトレーニングで除外されたタスクでモデルを微調整する。これを下流の各タスクについて繰り返す。この方法を「リーブオンアウト」マルチタスク

トレーニングと呼んでいます。この方法は、事前訓練で学習したモデルを、事前訓練で見ていないタスクで微調整するという現実の設定をシミュレートしています。マルチタスク事前学習は、教師ありタスクの多様な混合を提供することに留意する。他の分野（例えば、コンピュータビジョン（Oquab et al., 2014; Jia et al., 2014; Huh et al., 2016; Yosinski et al., 2014））では、事前訓練に教師ありデータセットを使用するので、我々は、マルチタスク事前訓練混合から教師なしタスクを省いても良い結果が得られるかどうかに興味があった。そこで、3つ目のバリエーションでは、$K = 2^{19}$で検討したすべての教師ありタスクの例-比例混合で事前学習を行うことにしました。これらすべてのバリエーションにおいて、2^{19}ステップの事前学習を行った後、2^{18}ステップの微調整を行うという標準的な手順を踏んでいる。

表12では、これらのアプローチの結果を比較する。比較のため、ベースライン（事前学習後に微調整）と、$K = 2^{19}$の例-比例混合に対する標準的なマルチタスク学習（微調整なし）の結果も示している。その結果、マルチタスクの事前学習後にファインチューニングを行うことで、ベースラインと同等の性能が得られることがわかりました。これは、マルチタスク学習後に微調整を行うことで、3.5.2節で説明した混合率の違いによるトレードオフを軽減できることを示唆している。興味深いことに

このことは、様々なタスクで学習したモデルでも、新しいタスクに適応できることを示唆している（つまり、マルチタスク事前学習は劇的なタスク干渉を引き起こさない可能性がある）。最後に、教師ありのマルチタスク事前訓練は、翻訳タスクを除くすべてのケースで著しく成績が悪化しました。このことは、翻訳タスクでは（英語の）事前トレーニングの恩恵が少なく、他のタスクでは教師なし事前トレーニングが重要な要素であることを示唆していると思われる。

3.6. スケーリング

機械学習研究の「苦い教訓」は、追加の計算を活用できる一般的な手法は、最終的に人間の専門知識に依存する手法に勝つと主張している（Sutton, 2019; Hestness et al., 2017; Shazeer et al., 2017; Jozefowicz et al., 2016; Mahajan et al., 2018; Shazeer et al. 2018, 2017; Huang et al., 2018b; Keskar et al., 2019a）．最近の結果は、これがNLPにおける転移学習にも当てはまる可能性を示唆している（Liu et al., 2019c; Radford et al., 2019; Yang et al., 2019; Lan et al., 2019）、すなわち、スケールアップすると、より慎重に設計された手法に比べて性能が向上することが繰り返し示されてきた。ただし、スケールアップには、より大きなモデルを使用する、より多くのステップについてモデルをトレーニングする、アンサンブルするなど、さまざまな方法が考えられます。このセクションでは、これらの方法を比較します。
4倍以上の計算能力を与えられた。どのように使うべきでしょうか?

ベースラインモデルから始めます。このモデルは220Mのパラメータを持ち、それぞれ2^{19}ステップと2^{18}ステップの事前訓練と微調整が施されています。エンコーダとデコーダは、「$\text{BERT}_{\text{BASE}}$」と同様のサイズになっています。モデルサイズの増加を実験するために、"$\text{BERT}_{\text{LARGE}}$ " Devlin et al. (2018)のガイドラインに従い、$d_{\text{ff}} = 4096$、$d_{\text{model}} = 1024$、$d_{\text{kv}} = 64$および
16層の注意機構次に、エンコーダとデコーダにそれぞれ16層と32層を持つ2つの変種を生成し、元のモデルの2倍と4倍のパラメータを持つモデルを作り出した。この2種類のモデルは、計算コストも2倍と4倍になる。
ベースラインと2つの大きなモデルを用いて、4倍の計算量を使用する3つの方法を検討した：4倍のステップ数でトレーニングする方法、2倍大きなモデルで2倍のステップ数でトレーニングする方法、4倍大きなモデルを「ベースライン」のトレー

ニングステップ数でトレーニングする方法です。
トレーニングのステップを増やすと、プリトレーニングとファインチューニングのステップの両方をスケールアップし
を簡略化しています。なお、事前学習ステップ数を増やすと、C4が非常に大きいため、2^{23}ステップの学習でも1回の通過で終わらないため、より多くの事前学習データを効果的に取り込んでいる。

モデルが4×多くのデータを見るための別の方法は、バッチサイズを4倍大きくすることです。これは、より効率的な並列化により、潜在的にトレーニングの高速化をもたらすことができます。しかし、4×大きなバッチサイズでトレーニングすると、4×多くのステップのトレーニングとは異なる結果が得られる可能性があります（Shallue et al.、2018）。これら2つのケースを比較するために、4×大きなバッチサイズでベースラインモデルを訓練する追加の実験を含めています。

私たちが検討している多くのベンチマークでは、モデルのアンサンブルを使用してトレーニングや評価を行うことで、さらなる性能を引き出すことが一般的に行われています。これにより、追加の計算を直交的に使用することができます。他のスケーリング方法とアンサンブルを比較するために、4つの別々に事前訓練され、微調整されたモデルのアンサンブルの性能も測定します。アンサンブル全体のロジットを平均化してから、出力ソフトマックス非線形に投入し、集約された予測値を得ます。4つのモデルを事前に訓練する代わりに

スケーリング戦略	グリュー	シーエヌエヌエム	スクワッド	スグレもの	エンデ	エヌフアール	エンロ
F ベースライン	83.28	19.24	80.88	71.36	26.98	39.82	27.65
1×サイズ、*4*×トレーニングステップ	85.33	19.33	82.45	74.72	27.08	40.66	27.93
*1*倍サイズ、*4*倍バッチサイズ	84.60	19.42	82.52	74.64	27.07	40.60	27.84
2×サイズ、*2*×トレーニングステップ	**86.18**	19.66	**84.18**	77.18	27.52	**41.03**	28.19
4×サイズ、1×トレーニングステップ	**85.91**	19.73	**83.86**	**78.04**	27.47	40.71	28.10
4× アンサンブル	84.77	**20.10**	83.09	71.74	**28.05**	40.53	**28.57**
4× アンサンブル、ファインチューンのみ	84.05	19.57	82.36	71.55	27.55	40.22	28.09

表13：ベースラインモデルをスケールアップする様々な方法の比較。微調整されたモデルのアンサンブルを除くすべての方法は、ベースラインと比較して4倍の計算を必要とする。「サイズ」はモデルのパラメータ数、「学習時間」は以下の通りです。

を、事前学習と微調整の両方に使用されるステップ数に置き換えます。

そのため、1つの訓練済みモデルを用いて、4つの微調整されたモデルを作成する方法が、より安価に利用できます。この方法では、4倍の計算予算全体を使用することはできませんが、他のスケーリング方法と同等の性能が得られるかどうかを確認するために、この方法を取り入れました。

のメソッドを使用しています。

これらの様々なスケーリング方法を適用した後に達成された性能は、表13に示されています。当然のことながら、トレーニング時間やモデルサイズを増やすと、ベースラインが一貫して改善されます。4倍のステップ数で学習する場合と、4倍のステップ数で学習する場合では、明確な勝敗はつきませんでした。とか、4倍以上のバッチサイズを使うとか、どちらも有効でしたが。一般的にモデルサイズを大きくしただけの場合と比較して、さらに性能が向上しています。訓練時間やバッチサイズ2倍大きなモデルを2倍長く訓練した場合と、4倍大きなモデルを訓練した場合では、調査したどのタスクでも大きな差は見られませんでした。このことは、訓練時間を長くしたり、モデルサイズを大きくしたりすることがパフォーマンスを向上させる補完的な手段であることを示唆しています。また、ア

ンサンブルは、スケールによって性能を向上させる直交的で効果的な手段を提供することが示唆された。いくつかのタスク（CNN/DM、WMT英語→ドイツ語、WMT英語→ルーマニア語）では、完全に別々に訓練された4つのモデルのアンサンブルは、他のすべてのスケーリングアプローチを大幅に上回った。また、事前に一緒に訓練されたモデルを別々に微調整するアンサンブルも、ベースラインよりも大幅に性能が向上しており、より安価に性能を向上させる手段を示唆しています。唯一の例外はSuperGLUEで、どちらのアンサンブルアプローチもベースラインを大幅に上回った。

スケーリング方法の違いにより、性能とは別のトレードオフがあることに注意する。例えば、より大きなモデルを使用すると、下流の微調整や推論がより高価になることがあります。一方、小さなモデルをより長く事前学習させるコストは、多くの下流タスクに適用される場合、効果的に償却されます。これとは別に、次のことに注意する。
*N個の*別々のモデルをアンサンブルすることは、N×高い計算コストを持つモデルを使用するのと同様のコストとなる。その結果、最終的なモデルの使用に対する何らかの配慮が
スケーリング方式を選択する際に重要なことです。

3.7. すべてをまとめる

私たちは現在、体系的な研究から得られた知見を活用し、一般的なNLPベンチマークでどこまで性能を押し上げることができるかを検討しています。また、大量のデータに対してより大きなモデルを学習させることで、NLPのための転移学習の現在の限界を探ることにも興味があります。まず、ベースラインの学習方法から始め、以下のような変更を加えます：

目的 私たちのベースラインのi.i.d.ノイズ除去目的を、SpanBERT (Joshi et al., 2019)に緩くインスパイアされた、セクション3.3.4で説明するスパンコラプション目的に置き換えた。具体的には、3の平均スパン長を使用し、元のシーケンスの15%を破損させる。我々は、この目的は、ターゲットシーケンス長が短いため、わずかに計算効率が高い一方で、わずかに優れたパフォーマンス（表7）を生成することを発見した。

長時間のトレーニング 私たちのベースラインモデルは、比較的少量の事前トレーニング（BERT（Devlin et al., 2018）と同じ$^{1}/_{4}$、XLNet（Yang et al., 2019）と同じ$^{1}/_{16}$、RoBERTa（Liu et al., 2019c）と同じ$^{1}/_{64}$、など）を使用しています。幸い、C4は十分な大きさであるため

は、データを繰り返すことなく、大幅に長い時間学習することができる（セクション3.4.2で示したように、これは不利になることがある）。セクション3.6で、事前学習を追加することが実際に有用であり、バッチサイズを大きくすることと学習ステップ数を増やすことの両方がこの利益をもたらすことを発見した。そこで、長さ512の^{211}個のシーケンスのバッチサイズに対して100万ステップの事前学習を行った。

1兆個の事前学習用トークン（ベースラインの約32倍）。セクション3.4.1では、RealNewsライク、WebTextライク、Wikipedia + TBCライクの事前学習で、1兆個のトークン（約32倍）を学習できることを示しました。

データセットは、いくつかの下流タスクにおいて、C4での事前学習を上回った。しかし、これらのデータセットのバリエーションは十分に小さく、1兆個のトークンを対象とした事前学習の過程で何百回も繰り返されることになる。セクション3.4.2で、この繰り返しが有害であることを示したので、代わりに

C4データセットを使い続けることを選択した。

モデルサイズ セクション3.6では、ベースラインモデルサイズをスケールアップすることで性能が向上することも示した。しかし、微調整や推論に利用できる計算資源が限られている環境では、より小さなモデルを使用することが有効である。これらの要因に基づき、我々は幅広いサイズのモデルを訓練する：

- **ベース。**これはベースラインモデルで、ハイパーパラメータはセクション3.1.1で説明する。およそ2億2千万個のパラメータを持つ。
- **小さい。を**使用してベースラインを縮小した、より小さなモデルを検討します。$d_{\text{model}} = 512$, $d_{\text{ff}} = 2{,}048$、8頭身の注目、で各6層のみ。
  エンコーダーとデコーダーを搭載しています。このバリアントは約6,000万個のパラメータを持っています。
- **大きい。**ベースラインでは、$\text{BERT}_{\text{BASE}}$ サイズのエンコーダとデコーダを使用しているため、エンコーダとデコーダがともに同程度のサイズである変形例も検討します。
  と$\text{BERT}_{\text{LARGE}}$ の構造を採用しています。具体的には、$d_{\text{model}} = 1{,}024$、$d_{\text{ff}} = 4{,}096$、$d_{\text{kv}} = 64$、16ヘッドアテンション、エンコーダとデコーダでそれぞれ24層を使用し、約7億7千万個のパラメータを使用する変種である。
- **3Bと11Bです。より**大きなモデルを使用した場合にどのようなパフォーマンスが可能かをさらに探るために、さらに2つのバリエーションを検討しました。どちらのケースでも

d_{model} = 1024、24層のエンコーダーとデコーダー、d_{kv} = 128となります。3B」バリアントでは、32層アテンションでd_{ff} = 16,384を使用し、その結果、約
28億のパラメータ。「11B」では、d_{ff} = 65,536を使用し、128ヘッドのアテンションで、約110億のパラメータを持つモデルを作成しました。特にd_{ff} を大きくしたのは、Transformerのフィードフォワードネットワークのような大きな密な行列の乗算は、最新のアクセラレーター（私たちがモデルを訓練するTPUなど）が最も効率的であるためです。

マルチタスク事前学習 セクション3.5.3では、微調整の前に、教師なしタスクと教師ありタスクのマルチタスク混合で事前学習すると、教師なしタスクだけで事前学習するのと同じくらい効果があることを示した。これは、「MT-DNN」（Liu et al., 2015, 2019b）が提唱するアプローチです。また、微調整の間だけでなく、トレーニングの全期間にわたって「下流」の性能を監視できるという実用的な利点もあります。そこで、最終的な実験セットでは、マルチタスクの事前学習を使用しました。我々は、より長く訓練された大規模なモデルは、より小さな訓練データセットにオーバーフィットする可能性が高いため、より多くのラベルなしデータの割合から利益を得ることができるという仮説を立てている。しかし、セクション3.5.3の結果から、マルチタスク事前学習後に微調整を行うことで、ラベルなしデータの割合が最適でない場合に発生しうる問題のいくつかを軽減できることが示唆されていることにも留意する。これらの考え方に基づき、標準的な例題比例混合（セクション3.5.2で説明）を行う前に、以下の人工データセットサイズをラベルなしデータに置き換えた：Smallは71万、Baseは262万、Largeは866万、3Bは3350万、11Bは13300万である。また、すべてのモデルで、事前学習時にWMT英仏データセットとWMT英独データセットの有効データセットサイズを1M例に制限した。

GLUEとSuperGLUEの個別タスクでのファインチューニング これまで、GLUEと

SuperGLUEのファインチューニングを行う際には、各ベンチマークのデータセットを連結し、GLUEとSuperGLUEで1回ずつしかモデルのファインチューニングを行わないようにしました。この方法は、ロジカルに研究を簡略化することができますが、タスクごとにファインチューニングを行う場合と比較して、一部のタスクで若干の性能を犠牲にすることがわかりました。個々のタスクで微調整を行う場合、すべてのタスクを一度にトレーニングすることで軽減される可能性のある問題は、リソースの少ないタスクに素早く適合してしまう可能性があることです。例えば、211個の長さ512個の配列からなる大きなバッチサイズでは、リソースの少ないGLUEやSuperGLUEタスクの多くで、データセット全体が各バッチに複数回出現することになります。そこで、GLUEとSuperGLUEの各タスクのファインチューニングでは、より小さなバッチサイズである8個のlength-512シーケンスを使用しています。また、チェックポイントを5,000ステップごとではなく、1,000ステップごとに保存し、オーバーフィットする前にモデルのパラメータにアクセスできるようにしました。

ビームサーチ これまでの結果はすべて、貪欲なデコードを使用して報告されています。長い出力シーケンスを持つタスクでは、ビームサーチを使うことで性能が向上することがわかりました（Sutskever et al.、2014）。具体的には、WMT翻訳とCNN/DM要約タスクに対して、ビーム幅4、長さペナルティα=0.6（Wu et al.、2016）を使用する。

テストセット 今回の実験は最終セットなので、検証セットではなく、テストセットでの結果を報告する。CNN/DailyMailでは、標準的なテストセットを使用しました。

をデータセットに追加した。WMTタスクの場合、英独はnewstest2014、英仏はnewstest2015、英ルーマニアはnewstest2016を使用することに相当する。GLUE と SuperGLUE については、ベンチマーク評価サーバーを使用して公式テストセットのスコアを計算しました[15,16] SQuAD については、テストセットで評価するにはベンチマークサーバーで推論を実行する必要があります。残念ながら、このサーバーの計算リソースは、最大のモデルから予測値を得るには不十分です。そのため、SQuADの検証セットでの性能を報告することにしました。幸い、SQuADテストセットで最も高い性能を示したモデルは、検証セットでも結果を報告しているため、表向きは最先端であるモデルとの比較が可能です。

上記の変更点を除けば、ベースラインと同じ学習手順とハイパーパラメータ（AdaFactorオプティマイザ、事前学習用の逆平方根学習率スケジュール、微調整用の一定学習率、ドロップアウト正規化、語彙など）を使用している。参考までに、これらの詳細はセクション2に記述されている。

この最終的な実験セットの結果は、表14に示すとおりである。全体として、検討した24のタスクのうち、18のタスクで最先端の性能を達成した。予想通り、最大のモデル（110億パラメータ）は、すべてのタスクにおいて、モデルサイズのバリエーションで最も優れた性能を示しました。T5-3Bモデルは、いくつかのタスクで従来の技術水準を上回りましたが、モデルサイズを110億パラメータに拡大することが、最高の性能を達成するための最も重要な要素でした。次に、個々のベンチマークについて結果を分析します。

GLUEの平均スコアは90.3という最新鋭のスコアを達成しました。注目すべきは、自然言語推論タスクMNLI、RTE、WNLIにおいて、我々の性能がこれまでの最先端を大幅に上回ったことである。RTEとWNLIは、機械性能が歴史的に人間の性能に遅れをとっているタスクの2つで、それぞれ93.6と95.9である（Wang et al.、2018）。パラメータ数の点で、私たちの11Bモデルバリアントは、GLUEベンチマークに提出された最も大きなモデルです。しかし、ベストスコアの提出物のほとんどは、予測を生成するために大量のアンサンブルと計算を使用しています。例えば、ALBERTの最高成績のバリアント（Lan et al., 2019）は、我々の3Bバリアントと同様のサイズとアーキテクチャのモデルを使用しています（ただし、巧妙なパラメータ共有により、パラメータは劇的に少なくなっています）。GLUEでその印象的なパフ

ォーマンスを生み出すために、ALBERTの作者はタスクに応じて「6から17」のモデルをアンサンブルした。このため、ALBERT アンサンブルを用いた予測は、T5-11B を用いた予測よりも計算量が多くなることが予想されます。

SQuADについては、Exact Matchスコアで1ポイント以上、以前の最先端（ALBERT (Lan et al., 2019)）を上回りました。SQuADは3年以上前に作成された長年のベンチマークであり、直近の改良では、最先端を数パーセントポイントしか上回っていません。テストセットで結果が報告される場合、それらは通常、モデルのアンサンブルに基づくか、または外部データセット（例えば、TriviaQA（Joshi et al.、2017）またはNewsQA（Trischler et al.、2016））を利用して小さなSQuADトレーニングセットを補強しています。SQuADにおける人間のパフォーマンスは、82.30と推定されます。

Exact MatchとF1メトリックでそれぞれ91.22（Rajpurkar et al, 2016）であり、このベンチマークでさらに改善することに意味があるかどうかは不明です。

15. http://gluebenchmark.com
16. http://super.gluebenchmark.com

モデル	グリュー平均	CoLAマシューの	SST-2アキュラシー	MRPC F1	MRPCアキュラシー	STS-Bピアソン	STS-Bスピアマン
前ベスト	89.4[a]	69.2[b]	97.1[a]	**93.**[6b]	**91.**[5b]	92.7[b]	92.3[b]
T5-Small（スモール	77.4	41.0	91.8	89.7	86.6	85.6	85.0
T5-ベース	82.7	51.1	95.2	90.7	87.5	89.4	88.6
T5-ラージ	86.4	61.2	96.3	92.4	89.9	89.9	89.2
T5-3B	88.5	67.1	97.4	92.5	90.0	90.6	89.8
T5-11B	**90.3**	**71.6**	**97.5**	92.8	90.4	**93.1**	**92.8**

モデル	QQP F1	QQPアキュラシー	MNLI-mアキュラシー	MNLI-mmアキュラシー	QNLIアキュラシー	RTEアキュラシー	WNLIアキュラシー
前ベスト	74.8[c]	**90.**[7b]	91.3[a]	91.0[a]	**99.**[2a]	89.2[a]	91.8[a]
T5-Small（スモール	70.0	88.0	82.4	82.3	90.3	69.9	69.2
T5-ベース	72.6	89.4	87.1	86.2	93.7	80.1	78.8
T5-ラージ	73.9	89.9	89.9	89.6	94.8	87.2	85.6
T5-3B	74.4	89.7	91.4	91.2	96.3	91.1	89.7
T5-11B	**75.1**	90.6	**92.2**	**91.9**	96.9	**92.8**	**94.5**

モデル	SQuADスクワッドCB	SQuADスクワッドCB	SQuAD COPA	スーパーグルーブールQ
	ゆうようびせいぶつぐん	F1	平均	アキュラシー	F1	アキュラシー	アキュラシー
前ベスト	90.1[a]	95.5[a]	84.6[d]	87.1[d]	90.5[d]	95.2[d]	90.6[d]
T5-Small（スモール	79.10	87.24	63.3	76.4	56.9	81.6	46.0
T5-ベース	85.44	92.08	76.2	81.4	86.2	94.0	71.2
T5-ラージ	86.66	93.79	82.3	85.4	91.6	94.8	83.4
T5-3B	88.53	94.95	86.4	89.9	90.3	94.4	92.0
T5-11B	**91.26**	**96.22**	**88.9**	**91.2**	**93.9**	**96.8**	**94.8**

モデル	マルチRCエフワンエーBLEU	マルチRCゆうようびせいぶつぐんBLEU	ReCoRD F1	ReCoRDアキュラシーBLEU	RTEアキュラシーROUGE-1	WiCアキュラシーROUGE-2	WSCアキュラシールージュ・エル
前ベスト	**33.**[8e]	**43.**[8e]		**38.**[5f]	43.47[g]	20.30[g]	40.63[g]
T5-Small（スモール	26.7	36.0		26.8	41.12	19.56	38.35
前ベスト	30.9	41.2	90.6[d]	28.0	90.0[d]	42.85[d]	69.9[d]	39.4[d]
T5-Small	32.0	41.5	56.3	28.1	55.4	42.73	66.9	39.56
T5-3B	31.8	42.6		28.2		42.72	21.02	39.94
T5-ベース	79.7	43.1	75.0	28.1	74.2	81.5	68.3	80.8
T5-11B	32.1	43.4	86.8		85.9	**43.52**	**21.55**	**40.69**
T5-ラージ	83.3	50.7				87.8	69.3	86.3
T5-3B	86.8	58.3	91.2		90.4	90.7	72.1	90.4
T5-11B	**88.1**	**63.3**	**94.1**		**92.4**	**92.5**	**76.9**	**93.8**
	WMT EnDe	WMT EnFr	WMT EnRo		CNN/DM	CNN/DM	CNN/DM

表14：調査したすべてのタスクに対するT5亜種の性能。Small、Base、Large、3B、11Bは、それぞれ6000万、2億2000万、7億7000万、30億、110億のパラメー

タを持つモデル構成を指す。各表の1行目には、そのタスクの最先端技術（2019年10月24日時点）を報告し、上付き文字はその出典を示し、参考文献はこのキャプションの最後に記載しています。すべての結果は、検証セットを使用するSQuADを除き、テストセットで報告されています。 [a](Lan et al., 2019) [b](Wang et al., 2019c) [c](Zhu et al., 2019) [d](Liu et al., 2019c) [e](Edunov et al., 2018) [f](Lample and Conneau, 2019) [g] (Dong et al., 2019)

SuperGLUEについては、最先端を大きく上回りました（平均スコア84.6（Liu et al.、2019c）から88.9へ）。SuperGLUEは、「現在の最先端システムの範囲を超えているが、ほとんどの大学教育を受けた英語話者が解決可能」（Wang et al., 2019b）であるタスクを含むように設計されました。人間のパフォーマンスである89.8（Wang et al.、2019b）にほぼ匹敵します。興味深いことに、読解タスク（MultiRCとReCoRD）では、我々は人間のパフォーマンスを大きく上回っており、これらのタスクに使用される評価指標が、機械が作った予測に偏っている可能性を示唆しています。一方、COPAとWSCでは、人間が100％の精度を達成しており、これは我々のモデルの性能を大きく上回っています。このことは、特に低リソース環境において、我々のモデルが完成させることが困難な言語タスクが残っていることを示唆しています。

WMT翻訳タスクのいずれにおいても、最先端の性能を達成することはできませんでした。これは、英語だけのラベルなしデータセットを使用したことが原因の1つかもしれません。また、これらのタスクで最も優れた結果のほとんどは、高度なデータ増強スキームであるバックトランスレーション（Edunovら、2018；Lample and Conneau、2019）を使用していることに留意する。低リソース英語からルーマニア語ベンチマークに関する最先端の技術も、クロスランゲージの教師なしトレーニングの追加形式を使用しています（Lample and Conneau, 2019）。我々の結果は、スケールと英語による事前訓練が、これらのより洗練された手法の性能に匹敵するには不十分である可能性を示唆しています。より具体的には、英語からドイツ語へのnewstest2014セットの最良の結果は、WMT 2018 (Edunov et al., 2018)のはるかに大きなトレーニングセットを使用しており、我々の結果との直接的な比較は困難である。

最後に、CNN/Daily Mailでは、ROUGE-2-Fスコアで有意な差しかないものの、最先端の性能を達成した。ROUGEスコアの向上は必ずしもまとまりのある要約に対応しないことが示されている（Paulus et al.、2017）。さらに、CNN/Daily Mailは抽象的な要約のベンチマークとして提起されているが、純粋に抽出的なアプローチでもうまくいくことが示されている（Liu, 2019）。また、最尤法で訓練された生成モデルは、反復的な要約を生成しやすいと議論されている（See et

al.、2017）。これらの潜在的な問題にもかかわらず、我々は、我々のモデルが首尾一貫した、ほぼ正しい要約を生成することを発見した。付録Cで、チェリー以外の検証セットの例をいくつか紹介します。

T5は、その強力な結果を達成するために、我々の実験的研究から得られた知見と前例のないスケールを組み合わせている。セクション3.6では、ベースラインモデルの事前学習量やサイズをスケールアップすることで、大きな効果が得られることを明らかにしました。このことから、私たちは、T5に導入した「非スケーリング」な変更が、T5の強力な性能にどれだけ貢献したかを測定することに興味を持ちました。そこで、最終的な実験として、T5とT6を比較しました。
は、以下の3つの構成で構成されている：1つ目は、^{235}≈34Bトークンで事前学習した標準的なベースラインモデル、2つ目は、約1兆トークン（つまりT5で使用した事前学習と同じ量）で代わりに学習したベースライン、これを「ベースライン-1T」と呼びます。
3つ目は、T5-Baseです。ベースライン-1TとT5-Baseの差は、T5を設計する際に行った「非スケーリング」の変更であることに注意してください。このように、これら2つのモデルの性能を比較することで、私たちの体系的な研究から得られた知見の影響を具体的に測定することができます。

これら3つのモデル構成の性能は表15に示す通りである。セクション3.6の結果と一致し、事前学習を追加することでベースラインよりも性能が向上することがわかる。それにもかかわらず、T5-Baseはすべてのダウンストリームタスクでbaseline-1Tを大幅に上回っている。このことは、T5の性能に寄与する要因はスケールだけではないことを示唆している。

モデル	グリュー	シーエヌエヌエム	スクワッド	スグレもの	エンデ	エヌフアール	エンロ
Fン ベースライン	83.28	19.24	80.88	71.36	26.98	39.82	27.65
ベースライン-1T	84.80	19.62	83.01	73.90	27.46	40.30	28.34
T5-ベース	**85.97**	**20.90**	**85.44**	**75.64**	**28.37**	**41.37**	**28.98**

表15：T5-Baseと本稿の残りの部分で使用したベースライン実験セットアップの性能比較。結果は検証セットで報告されています。「Baseline-1T」は、ベースラインモデルを1兆個で事前学習させた場合の性能である。
トークン（T5モデルバリアントで使用されたのと同じ数）ではなく、2^{35}≒34Bトークン（ベースラインで使用されたのと同じ数）が使用されました。

を成功させる。私たちは、大型モデルはそのサイズの拡大だけでなく、こうした非スケーリング的な要素からも利益を得ていると仮説を立てています。

4. リフレクション

私たちの体系的な研究を終え、私たちはまず、最も重要な発見をいくつか要約することで締めくくります。この結果は、どのような研究手段が有望であるか、あるいはそうでないかということについて、いくつかのハイレベルな視点を提供するものである。最後に、この分野をさらに発展させるための効果的なアプローチとして、私たちが考えるいくつかのトピックを概説します。

4.1. テイクアウェイ

Text-to-text我々のText-to-textフレームワークは、同じ損失関数とデコード手順を用いて、様々なテキストタスクに対して単一のモデルを訓練する簡単な方法を提供する。このアプローチは、抽象的要約のような生成タスク、自然言語推論のような分類タスク、さらにはSTS-Bのような回帰タスクにうまく適用できることを示した。そのシンプルさにもかかわらず、Text-to-Text

フレームワークはタスクに特化したアーキテクチャと同等の性能を持ち、最終的にスケールと組み合わせることで最先端の結果を得ることができることがわかりました。

アーキテクチャ NLPのための転移学習に関するいくつかの研究では、Transformerのアーキテクチャのバリエーションが検討されているが、我々は、元のエンコーダ-デコーダの形式が、我々のテキスト-テキストフレームワークで最もうまくいくことがわかった。エンコーダ-デコーダモデルは、「エンコーダのみ」（BERTなど）や「デコーダのみ」（言語モデル）のアーキテクチャに比べて2倍のパラメータを使用しますが、計算コストは同程度になります。また、エンコーダーとデコーダーのパラメーターを共有することで、パラメーターの総数を半減させながら、大幅な性能低下を招かないことを示しました。

教師なし目標 全体として、ランダムに破損したテキストを再構築するためにモデルを訓練する「ノイズ除去」目標のほとんどは、テキストからテキストへのセットアップで同様のパフォーマンスを示すことがわかった。その結果、教師なし事前学習がより計算効率が高くなるように、短いターゲットシーケンスを生成する目的を使用することを提案します。

データセット Common Crawlのウェブダンプからヒューリスティックにクリーニングされたテキストからなる「Colossal Clean Crawled Corpus」（C4）を導入しました。C4と

を使用するデータセットでは、ドメイン内のラベルなしデータで学習することで、いくつかの下流タスクの性能を向上させることができることを発見した。しかし、単一のドメインに限定すると、通常、データセットが小さくなる。また、ラベルなしデータセットが十分に小さく、事前学習の過程で何度も繰り返される場合、性能が低下する可能性があることを示した。このことは、一般的な言語理解タスクにC4のような大規模で多様なデータセットを使用する動機付けとなる。

訓練戦略 訓練済みのモデルのパラメータをすべて更新しながら微調整を行うという基本的なアプローチは、パラメータの更新を少なくするように設計された手法よりも優れていることがわかったが、すべてのパラメータを更新することは最もコストがかかる。また、複数のタスクに対して一度にモデルを学習させるための様々なアプローチも実験した。これは、テキストトゥテキストの設定では、バッチを構築する際に異なるデータセットの例を混合することに相当する。マルチタスク学習における最大の関心事は、各タスクをどのような割合で学習させるかを設定することである。我々は、教師なし事前学習と教師あり微調整という基本的なアプローチの性能に匹敵する混合比率を設定する戦略を最終的に見つけることができなかった。しかし、タスクの混合で事前学習を行った後に微調整を行うことで、教師なし事前学習と同等のパフォーマンスが得られることがわかりました。

より多くのデータでモデルをトレーニングする、より大きなモデルをトレーニングする、モデルのアンサンブルを使用するなど、追加された計算能力を利用するためのさまざまな戦略を比較しました。しかし、より多くのデータでより小さなモデルをトレーニングすることは、より大きなモデルをより少ないステップでトレーニングすることよりも優れていることが多いことがわかりました。また、モデルのアンサンブルは、単一のモデルよりも大幅に優れた結果を提供することができ、追加の計算を活用する直交的な手段を提供することを示しました。同じベースとなる事前学習済みモデルから微調整されたモデルのアンサン

ブルは、すべてのモデルを完全に別々に事前学習・微調整するよりも成績が悪かったが、微調整のみのアンサンブルは単一モデルを大幅に上回った。

限界への挑戦 以上のような知見を組み合わせ、大幅に大規模なモデル（最大110億パラメータ）を学習させることで、検討した多くのベンチマークで最先端の結果を得ることができました。教師なしトレーニングでは、C4データセットからテキストを抽出し、連続したトークンのスパンを破損させるノイズ除去の目的を適用しました。個々のタスクで微調整を行う前に、マルチタスク混合で事前トレーニングを行いました。全体として、我々のモデルは1兆個以上のトークンで訓練された。本成果の複製、拡張、応用を容易にするため、本コード、C4データセット、および各T5変種の事前訓練済みモデル重みを公開します[1]。

4.2. アウトルック

大型モデルの不都合 私たちの研究から得られた意外だが重要な結果は、大型モデルの方がより良いパフォーマンスを発揮する傾向があるということである。これらのモデルを実行するために使用されるハードウェアが絶えず安価で強力になっているという事実は、スケールアップがより良いパフォーマンスを達成するための有望な方法であり続ける可能性を示唆しています（Sutton, 2019）。しかし、例えば、次のような場合に、より小さい、またはより安価なモデルを使用することが有用な用途やシナリオが存在することは、常に変わりません。

クライアントサイドの推論や連合学習（Konečny` et al, 2015, 2016）。これに関連して、転移学習の有益な利用法として、低リソースタスクで優れた性能を達成する可能性がある。低リソースタスクは、より多くのデータをラベル付けするための資産がない環境で発生することが多い（定義上）。そのため、低リソースのアプリケーションでは、計算機へのアクセスが制限され、追加コストが発生することがよくあります。そのため、我々は、より安価なモデルでより高い性能を実現する手法の研究を提唱し、転移学習が最も影響力のある場所に適用できるようにします。この線に沿ったいくつかの現在の研究には、蒸留（Hinton et al., 2015; Sanh et al., 2019; Jiao et al., 2019）、パラメータ共有（Lan et al., 2019）、および条件付き計算（Shazeer et al., 2017）などがあります。

より効率的な知識抽出 事前学習の目的の一つは、（大雑把に言えば）下流タスクのパフォーマンスを向上させる汎用的な「知識」をモデルに提供することであることを思い出してください。本研究で用いた方法は、現在一般的に行われているもので、破損したテキストのスパンをノイズ除去するためにモデルを訓練するものである。この単純な手法は、モデルに汎用的な知識を教えるには、あまり効率的ではないのではないかと考えています。より具体的には、まず1兆個のテキストでモデルを訓練する必要なく、優れた微調整性能を達成できるようになれば便利です。これと並行して行われたいくつかの研究では、実際のテキストと機械で生成されたテキストを区別するためにモデルを事前に訓練することで効率を向上させている（Clark et al.、2020）。

タスク間の類似性の定式化 我々は、ラベルのないドメイン内データで事前学習することで、下流タスクのパフォーマンスが向上することを観察した（セクション3.4）。この発見は、SQuADがWikipediaからのデータを使って作成されたという事実のような基本的な観察にほとんど依存している。事前学習と下流タスクの間の「類似性」についてより厳密な概念を定式化することは、どのようなラベルなしデータを使用するかについてより原則的な選択を行う上で有用であろ

う。コンピュータビジョンの分野では、この線に沿った初期の経験的研究がいくつかあります（Huh et al., 2016; Kornblith et al., 2018; He et al.）タスクの関連性のより良い概念は、*教師ありの*事前学習タスクの選択にも役立ち、これはGLUEベンチマークに役立つことが示されている（Phang et al.、2018年）。

言語にとらわれないモデル 私たちが研究した翻訳タスクにおいて、英語のみの事前学習では最先端の結果が得られないことに失望しました。また、語彙がどの言語をエンコードできるかを事前に特定する必要があるため、ロジスティックな困難を回避することにも関心があります。これらの問題を解決するために、我々は言語に依存しないモデル、すなわちテキストの言語に関係なく、与えられたNLPタスクを優れたパフォーマンスで実行できるモデルをさらに調査することに興味があります。これは、世界の人口の大多数が英語を母国語としていないことを考えると、特に重要な問題である。

この論文の動機は、最近、NLPのための転移学習に関する研究が盛んに行われるようになったことである。私たちがこの研究を始める前に、これらの進歩により、学習ベースの手法がまだ有効であることが示されていなかった場面で、すでにブレークスルーが実現されていました。例えば、難易度の高いSuperGLUEベンチマークにおいて、人間レベルの性能とほぼ同等になるなど、この傾向を継続できることを嬉しく思う。

現代の伝達学習パイプラインのためのものです。私たちの成果は、わかりやすく統一されたテキスト-テキストフレームワーク、新しいC4データセット、そして体系的な研究からの洞察の組み合わせに起因しています。さらに、この分野の経験的な概要と、その立ち位置についての展望を提供しました。我々は、一般的な言語理解という目標に向けて、転移学習を用いた研究が継続されることを期待している。

謝辞

Grady Simon、 Noah Fiedel、 Samuel R． Bowman, Augustus Odena, Daphne Ippolito, Noah Constant, Orhan Firat, Ankur Bapna, Sebastian Ruderには、この原稿に対するコメント、Zak StoneとTFRCチームのサポート、Austin Tarangoにはデータセット作成に関する指導、 Melvin Johnson, Dima Lepikhin, Katrin Tomanek, Jeff Klingner, Naveen Arivazhaganにはマルチタスク機械翻訳への洞察を、 Neil Houlsbyにはアダプタ層に関するコメント、そして、 Oga WichowskaにはOla Spyra、 Michael Banfield、 Yi Lin、 Frank Chenにはインフラに関するサポートを、 Edienne PotにはNei-HoulsbyとTMのサポート、そして、Navi Houlzagan にはNei-Karanのコメントをいただいた；Olga Wichowska、Ola Spyra、Michael Banfield、Yi Lin、Frank Chenにはインフラストラクチャの支援を、Etienne Pot、Ryan Sepassi、Pierre RuyssenにはTensorFlow Datasetsのコラボレーションを、Rohan AnilにはCommon Crawlのダウンロードパイプラインの支援を、Robby NealeとTaku KudoにはSentencePieceを、その他Google Brainチームの多数のメンバーには議論や洞察をいただいた。

付録A.貢献度について

Colinはこのプロジェクトの範囲を設計し、この論文を書き、セクション3.1から3.6のすべての実験を実行し、コードベースの大部分を貢献した。Noamは、text-to-textフレームワーク、教師なし目的、データセットミキシング戦略を含む多くのアイデアを提供し、基本Transformerモデルとそのアーキテクチャのバリエーションを実装し、セクション3.7の実験を実行しました。Adamはこのプロジェクトのすべてのエンジニアリングを監督し、C4データセットを作成し、データセットパイプラインを実装し、様々なベンチマークデータセットを追加しました。Katherineは、実験の調整、ドキュメントの作成と更新、ベースラインの設計に役立つ実験の実行、コードベースの多くの部分への貢献を行った。Sharanは、必要なデータセットとプリプロセッサの一部を提供し、様々な予備実験を行い、さらにコードベースのオープンソース化を共同リードしました。MichaelはWinogradデータセットのすべての側面を所有し、私たちが使用するデータセットの多くを取り込み、私たちのインフラストラクチャにさまざまな改善や修正を提供し、いくつかの予備実験を実行しました。Yanqiは、合理的なベースラインを確立するための実験とメソッドの実装を行い、セクション3.7でモデルの最終的な微調整を手伝いました。Weiもまた、最終的な微調整を行い、プリプロセッサの一部を改良した。Peterは、初期バージョンの事前学習データセットを試作し、SQuADとCNN/DMタスクに関連する問題を解決した。すべての著者は、この研究で私たちがたどった範囲と研究の方向性を設定するのに貢献しました。

付録B．WNLIを当社のText-to-Textフォーマットへ変換する。

なお、セクション2.4で説明したように、WNLIからのデータで訓練は行っていない。その代わりに、WNLIテストセットで評価する場合（セクション3.7の結果について）、WSCとDPRで訓練したモデルを使用して評価できるように、WNLIテストセットを「参照名詞予測」テキスト-テキスト形式に変換します。我々のWNLIプリプロセッサは、Heら(2019)が提案したものに触発されている。WNLIの例は、前

提、仮説、NU仮説がTrueかFalseかを示すラベルで構成されていることを思い出してください。セクション2.4の例を用いると、仮説は "The city councilmen refused the demonstrators a permit because they fear violence. "となり、前提は "The demonstrators feared violence."、ラベルは "False "となります。まず、前提に含まれるすべての代名詞（この例では "they"）がどこにあるか調べます。次に、大文字と小文字、句読点を無視して、各代名詞の前後にある単語のうち、仮説の部分文字列（この例では「暴力を恐れた」）であるものの最大数を求める。前提に複数の代名詞候補が含まれている場合は、仮説の部分文字列のうち、最も大きな文字列が先行または後続する代名詞を選びます。そして、前提の代名詞をアスタリスクで囲んで強調する。候補の名詞（モデルの予測値と比較し、TrueまたはFalseのラベルを得る）については、仮説から一致する部分文字列を削除し、オプションで所有格でないものにします（結果として「the demonstrators」となります）。

付録C．CNN/デイリーメールでの予測例

我々のモデルが流暢な要約を生成していることを示すために、検証セットにおいて、我々の最良のモデル（T5-11B）によるいくつかのデコード例を、グランドトゥルースの要約と一緒に掲載しました。これらの例は無作為に選んだものであり、サクラを入れたわけではありません。

1. **真実:** 南アフリカのクルーガー国立公園で、ヒョウはトゲのある生き物と戦って引き下がらなかった。 野生動物愛好家リスル
   ヒョウとヤマアラシは茂みから転げ落ち、道端で戦い始めた。

   予測: ヒョウは、クルーガー国立公園（南アフリ カ）で、気難しいヤマアラシを昼食にしようとした。 しかし、トゲのある生き物が戦いの間に引き下がることを拒否したため、肉食動物はしっかりとその場所に置かれた。野生動物愛好家のリスル・ムールマン（41）は、道路に出かけ、カメラで武装しながらこの奇妙な戦いに遭遇した。

2. **地上の真実:** 研究者によると、家庭は未発見の種の温床であるという。
   ノースカロライナ州の40軒の住宅を調査したところ、8,000の細菌および古細菌の分類群が見つかった。
   .カリフォルニアの11の家は、私たちの間で非常に多様な菌類を明らかにした。

   予測: 研究者は、屋内バイオームは氷のない土地の約0.5%、約247,000平方マイルを占めると推定し、ほぼ同じ大きさです。
   ノースカロライナ州の40軒の家屋を対象とした研究では、ゲノム技術を使って8,000以上の細菌と古細菌の分類を記録した。
   また、カリフォルニア州の11軒の家屋を調査したところ、実にさまざまな菌類が発見されました。

3. **真実:** 月曜の夜、致死注射に使われる薬に欠陥があることが判明したため、州では死刑執行が一時的に停止された。ケリー・ギセンダナーは、致死量のペントバルビタールが濁って見えることが判明し、2度目の死刑延期となった。州当局はその決定について逡巡し、3度考えを変えてから死刑を執行しないことを決定している。
   ジョージア州司法長官サム・オレンズは、「憲法に則った方法で死刑が執行されることが不可欠だ」と述べた。

   予測: ジョージア州の死刑囚の中で唯一の女性であるケリー・ギセンダナーの死刑執行は、月曜日に2度目の延期となった。死刑執行チームは、致死量のペントバルビタールが濁って見えることを発見した。この濁った薬は、国内の他の地域で3回の死刑執行が失敗に終わった後、反対を表明している死刑反対派を後押しした。

4. **真実:** ダニ・アルベスはフランスとチリと対戦するブラジル代表に選ばれなかった。バルセロナのディフェンダーは、土曜日に人々にホットドッグを提供するところを撮影された。今週、アルベスは元チームメイトのジョゼ・ピントとチャリティーシングルをリリースした。
   この夏、パリ・サンジェルマンに移籍するとスペインで報道された。
   .

RAFFEL、SHAZEER、ROBERTS、LEE、NARANG、MATENA、ZHOU、LI、LIU

予想: ダニ・アルベスは、ブラジルのフランスとチリとの親善試合に選ばれなかった。

バルセロナの右サイドバックは、ホセ・ピントとチャリティーシングルをリリースした。

アルベスは、カウンターの後ろからスナックを提供するところを撮影された。

アルベスはまだヌーカンプでの新契約を提示されていません。

付録D．前処理された例

このセクションでは、私たちが検討した各データセットに対する私たちの前処理の例を提供します。

D.1. CoLA

オリジナル入力です：

センテンスです：ジョンはビルを自分の主人にした。

処理された入力：コーラ文：ジョンはビルを自分の主人にした。

オリジナルターゲット：1

加工対象：可

D.2. RTE

オリジナル入力です：

文1：ユーゴスラビアのイタリア人のうち、スロベニアに定住した人の割合は少ない（1991年の国勢調査で、スロベニアの住民約3000人がイタリア民族であると申告）。

文2：スロベニアの人口は3,000人です。

処理された入力：rte sentence1：ユーゴスラビアのイタリア人のうち、スロベニアに定住した人の割合は少ない（1991年の国勢調査で、スロベニアの住民約3000人がイタリア民族と申告）。 文2：スロベニアには3000人の住民がいる。

オリジナルターゲット：1

加工対象：not_entailment

D.3. エムエヌエルアイ

オリジナル入力です：

仮説です：セントルイス・カージナルスは常に勝利している。

前提：そうですね.....負けることは.....つまり、私はセントルイス出身で、セン

トルイス・カージナルスがいた頃は、ほとんど負けるチームでしたけど

処理された入力： MNLI仮説：仮説：セントルイス・カージナルスはいつも勝っている。前提：そうですね、負けるのは、私はセントルイス出身なので、セントルイス・カージナルスがいた頃は、ほとんど負けるチームだったんですけどね。

当初の目標： 2

加工対象： 矛盾

D.4. MRPC

オリジナル入力です：

文章1： 私たちが行動したのは、9月11日の経験というプリズムを通して、既存の証拠を新たな光で見たからです」ラムズフェルドは言った。

文2： むしろ、米国が行動したのは、政権が「9月の経験というプリズムを通して、既存の証拠を新しい光で見たからだ。
11 " .

処理された入力： mrpc sentence1：既存の証拠を見たから行動した
ラムズフェルドは、「9月11日の経験のプリズムを通して、新しい光の中で、」と述べた。 文2：むしろ、米国が行動したのは、政権が「9月11日の経験のプリズムを通して、新しい光の中で既存の証拠」を見たからである。

オリジナルターゲット： 1

加工対象： 同等品

D.5. キューエヌエルアイ

オリジナル入力です：

質問です： ジェベはどこで死んだのか?

センテンスです： チンギス・ハーンはその後すぐにスブタイをモンゴルに呼び戻し、ジェベはサマルカンドに戻る道中で死んだ。

処理された入力： qnli 質問：ジェベはどこで死んだのか? 文チンギスハンはスブタイをすぐにモンゴルに呼び戻し、ジェベはサマルカンドに戻る道中で死んだ。

オリジナルターゲット： 0

加工対象： エンタテインメント

D.6. QQP

オリジナル入力です：

Raffel、Shazeer、Roberts、Lee、Narang、Matena、Zhou、Li、Liu

質問1：古代ローマでは、どのような属性があれば、あなたは非常に好まれたのでしょうか？

質問2：フレッシャーズとしてIT企業に入社するためには、どうすればいいのでしょうか?

処理された入力：qqp question1：古代ローマでは、どのような属性があなたを高く評価したでしょうか? question2：フレッシャーズとしてIT企業に入社するために、OPPERTINUTYを得るには?

オリジナルターゲット： 0

処理対象： not_duplicate

D.7. エスティーエスツー

オリジナル入力です：

文： 技術的なノウハウを巧みに曲げて心理的な洞察に奉仕する映画人としてのフィンチャーの地位を確かなものにしている。

処理された入力： SST2 文：技術的なノウハウを芸術的に曲げて心理的な洞察に奉仕する映画製作者としてのフィンチャーの地位を確認する。
.

オリジナルターゲット： 1

加工対象： ポジティブ

D.8. エスティービー

オリジナル入力です：

文章1： ピュアチューンズの代表は、水曜日、すぐにコメントを得ることができませんでした。

文章2： ピュアチューンズの代表者は、この訴訟についてコメントするために木曜日に見つけることができませんでした。

処理された入力： stsb sentence1：ピュアチューンズの代表者は、水曜日、すぐにコメントを得ることができなかった。 文2：ピュアチューンズの代表者は
は、この訴訟に関するコメントを木曜日に得ることができませんでした。

当初の目標： 3.25

加工目標： 3.2

D.9. CB

オリジナル入力です：

仮説である： ヴァレンスが役立っていた
前提： 虚ろな脳を持つヴァレンス、高潔な付き人を持つヴァレンス。どうして

Raffel、Shazeer、Roberts、Lee、Narang、Matena、Zhou、Li、Liu
フィガーは、タイタニックの解剖学的な部分を自分で選んで軸にしたのですか? 彼は自分が助けていると思ったのだろうか?

処理された入力: CB仮説: ヴァレンスに助けられた 前提: ヴァレンス・ザ・ボイドブレインヴァレンス.....高潔な従者なぜフィガーは、タイタニック・アナトミーの自分の部分を選んで軸にすることができなかったのだろう? 彼は自分が助けているとでも思っているのだろうか。

オリジナルターゲット: 1

加工対象: 矛盾

D.10. コパ

オリジナル入力です：

質問： 効果

前提： 国家で政治的暴力が発生した。

選択肢1： 多くの市民が議事堂に移転した。

選択肢2： 多くの国民が他領土に避難した。

処理された入力： copa choice1：選択肢1：多くの市民が首都に移動した。選択肢2：多くの市民が他の領土に避難した：多くの国民が他の領土に避難した。 前提：国家で政治的暴力が発生した。 質問：効果

オリジナルターゲット： 1

ターゲットを加工したもの： 真

D.11. MultiRC

オリジナル入

力です：

答えてください： 伝統的な朝食の食べ物ではなく、パイしか食べるものがなかった

段落：送信1：昔々、ジョーイという名前のリスがいました。
送信2：ジョーイは外に出て、いとこのジミーと遊ぶのが大好きでした。
送信
3：ジョーイとジミーは一緒にくだらない遊びをして、いつも笑っていました。
銭 4：ある日、ジョーイとジミーはジュリーおばさんの池に一緒に泳ぎに行きました。
銭 5：ジョーイは朝早く起きて食べ物を食べにいきました。

Sent 6：彼はパイ以外に食べるものを見つけられなかった！
Sent
7：通常、ジョーイは朝食にシリアルや果物（梨）、オートミールを食べていました。
Sent 8：食べた後、彼とジミーは池に行きました。
Sent 9：
そこに行く途中、彼らは友達を見ました。

送信10：彼らは水に飛び込み、数時間泳ぎました。
送信11：太陽は出ていましたが、風は冷たかったです。
Sent 12：ジョーイとジミーは水から上がり、家まで歩き始めました。
Sent 13：彼らの毛は濡れていて、風は彼らを冷やしま

RAFFEL、SHAZEER、ROBERTS、LEE、NARANG、MATENA、ZHOU、LI、LIU
Sent 14: いつ
した。
Sent 15: ジョイは赤と緑のドットが入った青いシャツを着た。
Sent 16: 彼らは家に帰り、乾かし、ジミーはお気に入りの紫のシャツを着た:
2匹のリスはジョーイのお母さん、ジャスミンが作った餌を食べ、眠りにつきました。
。

質問です: ジョーイが朝起きて朝食を食べた時に驚いたのはなぜ?

処理された入力: multirc 質問です: ジョーイが朝起きて朝食を食べたとき、なぜ驚いたのか? 答えてください: パイしかなかったからです。

Sent 1: 昔々、ジョーイという名前のリスがいました。
Sent 2: ジョーイは外に出て、いとこのジミーと遊ぶのが好きでした。
Sent 3: ジョーイとジミーは一緒にくだらない遊びをして、いつも笑っていました。
Sent 4: ある日、ジョーイとジミーは一緒に泳ぎに出かけました。

Sent 5: ジョーイは朝早く起きて、出発する前に何か食べ物を食べようとしました。
Sent 6: 彼はパイ以外に食べるものを見つけられなかった!

Sent 7: 通常、ジョーイは朝食にシリアル、果物（梨）、オートミールを食べていた。
Sent 8: 食べた後、彼とジミーは池に行きました。
Sent 9: その途中、彼らは友達のジャックラビットを見ました。
Sent 10: 彼らは水に飛び込んで泳ぎました。

Sent 11: 太陽は出ていたが、風は冷たかった。
Sent 12: ジョーイとジミーは水から上がり、家まで歩き始めた。
Sent 13: 彼らの毛皮は濡れており、風は彼らを冷やした。
送信14: いつ。彼らは家に帰り、体を乾かし、ジミーはお気に入りの紫のシャツを着た。
送信15：ジョイは赤と緑のドットが入った青いシャツを着た。
送信16：その2匹のリスは、ジョーイのお母さん、ジャスミンが作った餌を食べて、寝てしまいました。

オリジナルターゲット： 1

ターゲットを加工したもの： 真

D.12. ワイシー

オリジナル入力です：

POSN

文1： 侮辱的だったのは、彼の行為の熟慮である。

文2： 陪審員の審議は.

単語： 熟慮

処理された入力： wic pos: N sentence1: It was the deliberation of his act that was insulting . sentence2: 陪審員の審議 ... word: 審議

オリジナルターゲット： 0

処理対象です： 偽

D.13. WSCとDPR

オリジナル入力です：

スパン2テキスト：

it **スパン1テキス**

ト： stable **スパン**

2インデックス： 20

スパン1のインデックス1

本文馬小屋はとても広く、4つの馬小屋があり、大きな窓から庭に出られるので、風通しが良く快適でした。

処理された入力： WSC: 馬小屋はとても広々としていて、4つの良い馬房があり、大きなスイングウィンドウが庭に面しているので、気持ちよく風通しが良い。

オリジナルターゲット： 1

加工対象： 安定

D.14. CNN/Daily Mail

Original input: マルアン・フェライニとアドナン・ヤヌザイは、単なるチームメイトではなく、ベストメイトであることを世界に示し続けている。マンチェスター・ユナイテッドとベルギーの二人は、水曜日のニューカッスル戦を前に、月曜日の夜、レストランに出かける写真を投稿した。ヤヌザイはフェライニと友人の真ん中で、まるでジャックソン5をテーマにした夜であるというメモを受け取らなかった人物のようにポーズを取っている。 プレミアリーグ
アドナン・ヤヌザイとマルアン・フェライニのデュオが、ダンスフロアで友人とポーズを決める
マンチェスター・ユナイテッドとベルギーのデュオ、フェライニとヤヌザイはピッチの内外で仲が良い。マンチェスター・ユナイテッドのエース、フェライニがベンチに駆け寄り、友人のヤヌザイとqpr戦のゴールを祝う。背景にあるディスコ効果が説に拍車をかけるが、ヤヌザイは気にしないようである。
この後、友人たちと一緒にダンスフロアでポーズをとる。
しかし、水曜日のニューカッスル戦に勝利すれば、ルイ・ファン・ガール監督に少なくともタッピングを残すことができる。
ヤヌザイとロビン・ファン・ペルシは、ウェストブロムでマンチェスター・ユナイテッドのファンの前で祝杯をあげた。 ヤヌザイは、マンチェスター・ユナイテッドのオランダ人監督ルイ・ファン・ガールから知恵をもらった。 ヤヌザイとフェライニは、ニューカッスル戦の前にダンスフロアで友人たちと一緒に踊る。

処理された入力： 要約：マルアン・フェライニとアドナン・ヤヌザイは、彼らが単なるチームメイトではなく、ベストメイトであることを世界に示し続けている。マンチェスター・ユナイテッドとベルギーの二人は、水曜日のニューカッスルとの試合を前に、月曜日の夜にレストランで外出する写真を投稿した。その中で、ヤヌザイはフェライニと友人の真ん中でポーズをとっており、まるでジャックソン5をテーマにした夜だというメモを受け取れなかった誰かのようだ。 プレミアリーグのデュオ、アドナン・ヤヌザイとマルアン・フェライニが友人とダンスフロアでポーズをとっている。
マンチェスター・ユナイテッドとベルギーのデュオ、フェライニとヤヌザイはピッチの内外で仲が良い。マンチェスター・ユナイテッドのエース、フェライニがベンチに駆け寄り、友人のヤヌザイとqpr戦のゴールを祝う。背景にあるディスコ効果が説に拍車をかけるが、ヤ

RAFFEL、SHAZEER、ROBERTS、LEE、NARANG、MATENA、ZHOU、LI、LIU)
ヌザイは気にしないようである。
この後、友人たちと一緒にダンスフロアでポーズをとる。
しかし、水曜日のニューカッスル戦に勝利すれば、ルイ・ファン・ガール監督に少なくともタッピングを残すことができる。
ヤヌザイとロビン・ファン・ペルシは、ウェストブロムでマンチェスター・ユナイテッドのファンの前でフェライニと一緒に祝杯をあげた。 ヤヌザイは知恵をもらった。

マンチェスター・ユナイテッドのオランダ人監督、ルイ・ファン・ハールから。 ジャヌザイとフェライニは、ニューカッスル戦の前に、友人たちと一緒にダンスフロアに繰り出します。

オリジナルターゲット： ベルギー人デュオは月曜の夜、友人たちとダンスフロアに繰り出した。 マンチェスター・ユナイテッドは水曜、プレミアリーグでニューカッスルと対戦する。レッドデビルズは、リーグ戦で7回目のアウェイ勝利を目指す。ルイ・ファンガール監督のチームは、現在4位のリバプールに2ポイント差をつけている。

加工対象： ベルギー人デュオは月曜の夜、友人たちとダンスフロアに繰り出した。 マンチェスター・ユナイテッドは水曜、プレミアリーグでニューカッスルと対戦する。レッドデビルズはリーグ戦で7回目のアウェイ勝利を目指す。ルイ・ファンガールのチームは現在4位のリバプールに2ポイント差をつけている。

D.15. スクワッド

オリジナル入力です：

質問です： 患者さんの肺の酸素濃度が上がると、何が変位するのでしょうか?

コンテクストです： 高気圧（高圧）医療では、特殊な酸素室を使用する
一酸化炭素中毒、ガス壊疽、減圧症（潜水病）などを治療するために、患者や医療スタッフの周囲の酸素分圧を高める装置です。一酸化炭素中毒、ガス壊疽、減圧症（潜水病）などがこの装置で治療されることがあります。肺の酸素濃度を高めると、ヘモグロビンのヘム基から一酸化炭素を置換することができます。酸素ガスはガス壊疽の原因となる嫌気性細菌にとって毒であるため、その分圧を高めることで細菌を殺すことができる。
減圧症は、潜水後、急激に減圧するダイバーに起こる。
ダイビングをすると、血液中に窒素やヘリウムなどの不活性ガスの気泡ができる。一刻も早く0 2の圧力を上げることが治療の一部となります。

処理された入力： 質問：患者の肺の酸素濃度を高めると、何が変位するのか? 文脈は? 高気圧（高圧）医療では、特殊な器具を使用します。
一酸化炭素中毒、ガス壊疽、減圧症（潜水病）などを治療するために、患者や医療スタッフの周囲に酸素分圧を高める酸素ボンベを設置します。一酸化炭素中毒、ガス壊疽、減圧症（潜水病）などがこの装置で治療されることがあります。肺の酸素濃度を高めると、ヘモグロビンのヘム基から一酸化炭素を置換することができます。酸素ガスはガス壊疽の原因となる嫌気性細菌にとって毒であるため、その分圧を高めることで細菌を死滅させるこ

RAFFEL、SHAZEER、ROBERTS、LEE、NARANG、MATENA、ZHOU、LI、LIU
とができます。減圧症は、潜水後の減圧が速すぎて、血液中に窒素やヘリウムなどの不活性ガスの気泡が発生した場合に起こります。できるだけ早く02の圧力を上げることが治療の一部となります。

オリジナルターゲット：一酸化炭素

加工対象：一酸化炭素

D.16. WMT 英語からドイツ語

原文入力です："ルイジはよく私に、兄弟が裁判になるのは絶対に嫌だと言っていました。"と書いています。

処理済み入力：英語からドイツ語への翻訳：「ルイジはよく私に、兄弟が裁判になるのは絶対に嫌だと言っていました」と彼女は書いています。

原文ママ："Luigi sagte oft zu mir, dass er nie wollte, dass die Brüder vor Gericht landen", schrieb sie.

処理対象：「Luigi sagte oft zu mir, dass er nie wollte, dass die Brüder vor Gericht landen", schrieb sie.

D.17. WMT 英語からフランス語へ

オリジナルの入力です：この画像は、スピッツァー望遠鏡が赤外線で記録したもので、数え切れないほどの世代の星の「家族の肖像画」を示しています。
が青い点として見え、さらに識別が難しいのは、星の分娩室にいるピンク色の「新生児」である。

処理された入力：英語からフランス語に翻訳する：この画像は、スピッツァー望遠鏡が赤外線で記録したもので、数え切れないほどの世代の星の「家族の肖像画」を示しています。

オリジナル・ターゲットスピッツァー望遠鏡が撮影した赤外線写真から、数え切れないほど多くの世代のエトワールの「家族の肖像」を見ることができます。
を識別するために、ユニバーサルの出産サロンで「新参者」と呼ばれています。

処理対象：スピッツァー望遠鏡が撮影した赤外線写真から、数え切れないほど多くの世代のエトワールの「家族の肖像」を見ることができます。
を識別するために、ユニバーサルの出産サロンで「新参者」と呼ばれています。

D.18. WMT 英語からルーマニア語

オリジナルの入力です：タコベルは、2022年までに米国で2,000店舗を増やす計画だという。

処理された入力：英語をルーマニア語に翻訳する：タコベルは、2022年までに米国内で2,000店舗を増やす予定と発表した。

当初の目標タコベルは、2022年にSUAで2000店舗の撤退を目指すことを表明しています。

加工対象： タコベルは、2022年にSUAで2000店舗の撤退を目指すと発表した。

RAFFEL、SHAZEER、ROBERTS、LEE、NARANG、MATENA、ZHOU、LI、LIU

付録E．全実験の各タスクのスコア

以下の表は、3.2〜3.6節で説明した実験において、すべてのタスクで達成されたスコアの一覧である。

			GLUE																SuperGLUE								WMT		
		スコア	CoLA	SST-2	MRPC	MRPC	エスティービー	エスティービーエスシーシー	QQP	QQP	エムエヌエルアイエム	MNLImm	キューエヌエルアイアック	RTE	CNN/DM			スクワッド		スコア	ブールキュー	CB	CB	コパ	マルチRC	マルチRC	ReCoRD	ReCoRD	RTE	ワイシー	せかいざひょうけい	エンデ	エヌファール	エンロ
テーブル	実験	平均	エムシーシー	アック	F1	アック	PCC	エスシーシー	F1	アック	アック	アック	アック	アック	R-1-FR-2-F		R-L-F	EM	F1	平均	アック	F1	アック	アック	F1	EM	F1	ゆうようびせいぶつくん	アック	アック	アック	BLEU	BLEU	BLEU
1	F ベースライン平均	83.28	53.84	92.68	92.07	88.92	88.02	87.94	88.67	91.56	84.24	84.57	90.48	76.28	41.33	19.24	38.77	80.88	88.81	71.36	76.62	91.22	91.96	66.20	66.13	25.78	69.05	68.16	75.34	68.04	78.56	26.98	39.82	27.65
1	ベースラインの標準偏差	0.235	1.111	0.569	0.729	1.019	0.374	0.418	0.108	0.070	0.291	0.231	0.361	1.393	0.065	0.065	0.058	0.343	0.226	0.416	0.365	3.237	2.560	2.741	0.716	1.011	0.370	0.379	1.228	0.850	2.029	0.112	0.090	0.108
1	事前トレーニングなし	66.22	12.29	80.62	81.42	73.04	72.58	72.97	81.94	86.62	68.02	67.98	75.69	58.84	39.19	17.60	36.69	50.31	61.97	53.04	65.38	71.61	76.79	62.00	59.10	0.84	20.33	17.95	54.15	54.08	65.38	25.86	39.77	24.04
2	F エンコード／デック、デノイズ	83.28	53.84	92.68	92.07	88.92	88.02	87.94	88.67	91.56	84.24	84.57	90.48	76.28	41.33	19.24	38.77	80.88	88.81	71.36	76.62	91.22	91.96	66.20	66.13	25.78	69.05	68.16	75.34	68.04	78.56	26.98	39.82	27.65
2	エンコード/デック、共有、デノイズ	82.81	55.24	91.86	91.58	88.24	87.43	87.58	88.69	91.60	83.88	84.01	90.23	73.65	41.11	18.78	38.48	80.63	88.49	70.73	77.13	95.04	96.43	65.00	66.16	22.98	68.95	68.09	70.76	68.18	75.96	26.72	39.03	27.46
2	エンコード/デコード、6層、ノイズ除去	80.88	46.26	92.09	91.51	87.99	87.01	86.76	87.93	90.97	82.20	82.41	88.83	71.48	40.83	18.97	38.31	77.59	86.07	68.42	73.79	91.70	92.86	67.00	61.02	19.62	61.26	60.33	72.20	65.99	75.00	26.38	38.40	26.95
2	言語モデル、デノイズ	74.70	24.50	90.60	86.08	78.92	85.22	85.42	85.40	88.99	76.72	77.05	86.02	64.62	39.49	17.93	36.91	61.14	71.37	55.02	65.47	60.08	71.43	58.00	43.03	2.94	53.35	52.31	53.07	58.62	63.46	25.09	35.28	25.86
2	プリフィクスLM、ノイズ除去	81.82	49.99	92.43	91.43	88.24	87.20	86.98	88.41	91.39	82.32	82.93	88.71	74.01	40.46	18.61	37.90	78.94	87.31	68.11	75.50	93.37	91.07	60.00	63.43	21.20	65.03	64.11	71.48	65.67	73.08	26.43	37.98	27.39
2	Enc/dec、LM	79.56	42.03	91.86	91.64	88.24	87.13	87.00	88.21	91.15	81.68	81.66	88.54	65.70	40.67	18.59	38.13	76.02	84.85	64.29	72.23	85.74	89.29	57.00	60.53	16.26	59.28	58.30	65.34	64.89	70.19	26.27	39.17	26.86
2	エンコード／デコード、シェア、LM	79.60	44.83	92.09	90.20	85.78	86.03	85.87	87.77	91.02	81.74	82.29	89.16	65.34	40.16	18.13	37.59	76.35	84.86	63.50	70.49	91.41	87.50	55.00	60.21	16.89	57.83	56.73	63.54	63.48	70.19	26.62	39.17	27.05
2	エンコード/デコード、6層、LM	78.67	38.72	91.40	90.40	86.52	86.82	86.49	87.87	91.03	80.99	80.92	88.05	65.70	40.29	18.26	37.70	75.32	84.06	64.06	71.38	85.25	89.29	60.00	57.56	16.79	55.22	54.30	66.79	63.95	71.15	26.13	38.42	26.89
2	言語モデル、LM	73.78	28.53	89.79	85.23	78.68	84.22	84.00	84.88	88.70	74.94	75.77	84.84	58.84	38.97	17.54	36.37	53.81	64.55	56.51	64.22	59.92	71.43	64.00	53.04	1.05	46.81	45.78	58.84	56.74	69.23	25.23	34.31	25.38
2	プリフィクス LM、LM	79.68	41.26	92.09	90.11	86.27	86.82	86.32	88.35	91.35	81.71	82.02	89.04	68.59	39.66	17.84	37.13	76.87	85.39	64.86	71.47	93.37	91.07	57.00	58.67	16.89	59.25	58.16	64.26	66.30	71.15	26.28	37.51	26.76
4	プレフィックスを用いた言語モデリング	80.69	44.22	93.00	91.68	88.48	87.20	87.18	88.39	91.41	82.66	83.09	89.29	68.95	40.71	18.94	38.15	77.99	86.43	65.27	73.55	83.95	87.50	55.00	59.65	18.89	61.76	60.76	68.59	65.67	73.08	26.86	39.73	27.49
4	BERT式（デブリンら、2018年）	82.96	52.49	92.55	92.79	89.95	87.68	87.66	88.47	91.44	83.60	84.05	90.33	75.45	41.27	19.17	38.72	80.65	88.24	69.85	76.48	94.37	94.64	61.00	63.29	25.08	66.76	65.85	72.20	69.12	75.00	26.78	40.03	27.41
4	デスハフリング	73.17	22.82	87.16	86.88	81.13	84.03	83.82	86.38	89.90	76.30	76.34	84.18	58.84	40.75	18.59	38.10	67.61	76.76	58.47	69.17	63.70	78.57	56.00	59.85	12.70	45.52	44.36	57.04	64.89	68.27	26.11	39.30	25.62
5	BERT式（デブリンら、2018年）	82.96	52.49	92.55	92.79	89.95	87.68	87.66	88.47	91.44	83.60	84.05	90.33	75.45	41.27	19.17	38.72	80.65	88.24	69.85	76.48	94.37	94.64	61.00	63.29	25.08	66.76	65.85	72.20	69.12	75.00	26.78	40.03	27.41
5	MASSスタイル（Song et al, 2019）	82.32	47.01	91.63	92.53	89.71	88.21	88.18	88.58	91.44	82.96	83.67	90.02	77.26	41.16	19.16	38.55	80.10	88.07	69.28	75.08	84.98	89.29	63.00	64.46	23.50	66.71	65.91	72.20	67.71	78.85	26.79	39.89	27.55
5	F 破損したスパンを置き換える	83.28	53.84	92.68	92.07	88.92	88.02	87.94	88.67	91.56	84.24	84.57	90.48	76.28	41.33	19.24	38.77	80.88	88.81	71.36	76.62	91.22	91.96	66.20	66.13	25.78	69.05	68.16	75.34	68.04	78.56	26.98	39.82	27.65
5	破損したトークンを落とす	84.44	60.04	92.89	92.79	89.95	87.28	86.85	88.56	91.54	83.94	83.92	90.74	79.42	41.27	19.31	38.70	80.52	88.28	68.67	75.90	96.02	94.64	56.00	65.06	23.92	65.54	64.60	71.12	67.40	74.04	27.07	39.76	27.82
6	汚職率 = 10	82.82	52.71	92.09	91.55	88.24	88.19	88.15	88.47	91.40	83.50	84.51	90.33	75.45	41.05	19.00	38.53	80.38	88.36	69.55	74.98	92.37	92.86	62.00	66.04	24.66	67.93	67.09	70.76	67.24	75.96	26.87	39.28	27.44
6	F 汚職率 = 15	83.28	53.84	92.68	92.07	88.92	88.02	87.94	88.67	91.56	84.24	84.57	90.48	76.28	41.33	19.24	38.77	80.88	88.81	71.36	76.62	91.22	91.96	66.20	66.13	25.78	69.05	68.16	75.34	68.04	78.56	26.98	39.82	27.65
6	汚職率 = 25	83.00	53.47	93.00	92.44	89.46	87.36	87.36	88.68	91.53	84.44	84.15	90.77	74.01	41.69	19.54	39.14	80.96	88.61	70.48	76.39	93.02	92.86	68.00	65.46	24.66	68.20	67.39	73.65	67.87	72.12	27.04	39.83	27.47
6	汚職率 = 50	81.27	46.26	91.63	91.11	87.99	87.87	87.64	88.70	91.57	83.64	84.10	90.24	70.76	41.51	19.32	38.89	79.80	87.76	70.33	75.02	93.05	92.86	68.00	62.97	24.13	64.94	64.13	72.20	68.50	77.88	27.01	39.90	27.49
7	F ベースライン（i.i.d.）	83.28	53.84	92.68	92.07	88.92	88.02	87.94	88.67	91.56	84.24	84.57	90.48	76.28	41.33	19.24	38.77	80.88	88.81	71.36	76.62	91.22	91.96	66.20	66.13	25.78	69.05	68.16	75.34	68.04	78.56	26.98	39.82	27.65
7	平均スパン長 = 2	83.54	53.82	92.20	93.05	90.44	87.85	87.71	88.42	91.40	84.28	84.46	90.88	77.62	41.23	19.39	38.69	82.09	89.69	72.20	77.06	90.43	91.07	70.00	66.28	26.13	71.34	70.61	75.45	68.34	78.85	26.76	39.99	27.63
7	平均スパン長 = 3	83.49	53.90	92.43	92.25	89.46	87.49	87.53	88.72	91.51	84.85	84.84	90.99	77.26	41.50	19.62	38.94	81.84	89.66	72.53	76.85	94.37	94.64	70.00	67.64	28.75	70.84	69.90	74.73	67.71	77.88	26.86	39.65	27.62
7	平均スパン長 = 5	83.40	52.12	93.12	92.63	89.71	88.70	88.47	88.84	91.64	84.32	84.29	90.79	76.90	41.39	19.24	38.82	82.05	89.79	72.23	77.06	83.06	89.29	69.00	68.16	30.12	71.36	70.53	75.81	69.91	79.81	26.88	39.40	27.53
7	平均スパン長 = 10	82.85	50.11	92.09	91.95	88.97	88.45	88.22	88.86	91.63	84.34	84.28	91.07	76.17	41.38	19.33	38.80	81.84	89.39	70.44	76.45	87.40	89.29	65.00	66.87	29.59	69.82	68.94	72.56	67.55	75.96	26.79	39.49	27.69
8	F C4	83.28	53.84	92.68	92.07	88.92	88.02	87.94	88.67	91.56	84.24	84.57	90.48	76.28	41.33	19.24	38.77	80.88	88.81	71.36	76.62	91.22	91.96	66.20	66.13	25.78	69.05	68.16	75.34	68.04	78.56	26.98	39.82	27.65
8	C4、アンフィルタード	81.46	48.01	91.63	92.72	89.95	87.79	87.60	88.31	91.27	82.30	82.34	88.71	72.20	41.09	19.14	38.54	78.78	87.04	68.04	75.75	89.17	91.07	62.00	65.52	25.60	62.42	61.58	69.68	67.08	72.12	26.55	39.34	27.21
8	リアルニュースのような	83.83	56.55	92.66	92.06	88.97	87.71	87.37	88.51	91.49	84.35	84.46	90.61	78.34	41.38	19.23	38.84	80.39	88.50	72.38	77.00	93.09	94.64	66.00	65.92	23.82	74.56	73.72	75.81	66.61	80.77	26.75	39.90	27.48
8	ウェブテキスト的	84.03	56.38	93.12	92.31	89.22	88.69	88.68	88.65	91.56	84.70	84.84	90.83	77.62	41.23	19.31	38.70	81.42	89.15	71.40	76.88	83.08	89.29	66.00	64.10	24.24	72.24	71.36	75.45	68.03	82.69	26.80	39.74	27.59
8	ウィキペディア	81.85	45.53	92.32	91.67	88.24	85.62	86.40	88.37	91.34	82.61	83.25	90.96	77.26	41.39	19.31	38.81	81.29	89.18	68.01	76.12	56.03	80.36	67.00	65.01	25.92	69.03	68.06	74.73	67.08	76.92	26.94	39.69	27.67
8	ウィキペディア + TBC	83.65	55.53	92.78	92.41	89.22	86.67	86.27	89.47	92.29	84.38	83.45	91.94	76.90	41.22	19.28	38.67	82.08	89.70	73.24	76.22	95.40	92.86	69.00	51.59	50.93	69.53	68.51	77.62	66.93	81.73	26.77	39.63	27.57
9	F フルデータセット	83.28	53.84	92.68	92.07	88.92	88.02	87.94	88.67	91.56	84.24	84.57	90.48	76.28	41.33	19.24	38.77	80.88	88.81	71.36	76.62	91.22	91.96	66.20	66.13	25.78	69.05	68.16	75.34	68.04	78.56	26.98	39.82	27.65
9	2^{29} (64回繰り返し)	82.87	53.82	92.78	91.79	88.73	87.56	87.58	88.73	91.54	84.07	84.21	90.59	73.65	41.18	19.19	38.67	80.97	88.90	72.03	76.76	92.96	92.86	66.00	65.11	26.76	69.35	68.49	75.81	67.24	82.69	26.83	39.74	27.63
9	2^{27} (256回繰り返し)	82.62	50.60	92.32	92.07	88.73	87.83	87.60	88.65	91.54	83.43	84.37	90.12	75.81	41.24	19.20	38.70	79.78	87.63	69.97	75.29	93.42	91.07	63.00	61.82	23.61	66.27	65.39	73.65	66.30	80.77	27.02	39.71	27.33
9	2^{25} (1,024回繰り返し)	79.55	43.84	91.28	89.32	85.05	85.92	85.74	88.05	91.09	81.29	81.72	87.90	69.31	40.66	18.57	38.13	76.27	84.58	64.76	72.63	83.97	82.14	64.00	59.39	17.94	56.94	56.04	64.98	65.20	73.08	26.38	39.56	26.80
9	2^{23} (4,096回繰り返し)	76.34	32.68	89.45	89.84	86.03	83.49	83.42	87.18	90.61	77.80	78.69	85.47	64.62	40.16	18.33	37.66	70.92	80.20	59.29	69.85	73.48	73.21	56.00	57.66	14.38	46.69	45.79	59.57	65.05	68.27	26.37	38.84	25.81
10	F 全パラメーター	83.28	53.84	92.68	92.07	88.92	88.02	87.94	88.67	91.56	84.24	84.57	90.48	76.28	41.33	19.24	38.77	80.88	88.81	71.36	76.62	91.22	91.96	66.20	66.13	25.78	69.05	68.16	75.34	68.04	78.56	26.98	39.82	27.65
10	アダプター層、d = 32	80.52	45.33	91.63	90.59	86.76	88.38	88.06	86.99	90.26	83.63	83.94	90.72	67.15	34.50	15.08	32.15	79.32	87.70	60.40	65.32	50.87	73.21	52.00	58.61	19.41	65.50	64.58	62.09	64.58	73.08	13.84	17.88	15.54
10	アダプター層、d = 128	81.51	45.35	92.89	91.49	88.24	87.73	87.65	87.73	90.93	83.64	84.09	90.52	72.56	36.71	16.62	34.37	79.47	87.61	63.03	69.20	52.21	75.00	56.00	61.08	18.05	67.94	66.97	68.59	66.77	73.08	19.83	27.50	22.63
10	アダプター層、d = 512	81.54	44.25	93.35	91.00	87.25	88.74	88.44	88.02	91.15	83.08	83.80	89.62	74.37	38.63	17.78	36.25	79.18	87.32	64.30	73.18	59.86	71.43	56.00	62.94	18.57	66.56	65.74	70.76	67.87	74.04	23.45	33.98	25.81
10	アダプター層、d = 2048	82.62	49.86	92.55	91.30	87.99	88.46	88.35	88.36	91.40	83.63	83.18	90.66	76.53	39.44	18.30	37.06	79.40	87.36	68.61	74.53	88.00	91.07	58.00	61.10	18.89	66.73	66.06	73.29	71.16	75.96	25.64	36.92	26.93
10	徐々に凍結を解除する	82.50	51.74	91.97	92.61	89.71	87.27	86.90	88.26	91.35	83.42	83.49	89.71	75.09	40.88	18.95	38.40	79.17	87.30	70.79	75.51	93.09	94.64	70.00	62.03	21.51	65.69	64.79	72.92	69.12	77.89	26.71	39.02	26.93
11	F ベースライン（プレトレーニング/ファインチューニング）	83.28	53.84	92.68	92.07	88.92	88.02	87.94	88.67	91.56	84.24	84.57	90.48	76.28	41.33	19.24	38.77	80.88	88.81	71.36	76.62	91.22	91.96	66.20	66.13	25.78	69.05	68.16	75.34	68.04	78.56	26.98	39.82	27.65
11	イコール	76.13	39.47	90.94	82.90	75.74	78.83	78.44	86.45	89.71	82.08	82.92	90.13	59.93	40.95	19.02	38.39	76.51	85.61	63.37	73.06	82.37	83.93	65.00	60.89	17.52	60.51	59.70	61.01	60.03	65.38	23.89	34.31	26.78
11	例-比例型、$K = 2^{16}$	80.45	42.07	91.97	90.97	87.50	85.41	85.04	86.89	90.10	83.01	83.66	90.74	72.56	41.16	19.04	38.59	77.25	85.72	69.95	76.67	86.38	89.29	70.00	65.93	27.91	62.78	61.95	76.90	65.83	73.08	24.35	34.99	27.10
11	例-比例型、$K = 2^{17}$	81.56	47.35	91.40	91.55	88.24	86.15	85.93	86.94	90.06	82.76	84.12	90.79	75.09	41.06	19.12	38.47	77.00	85.87	67.91	77.89	77.54	85.71	57.00	67.78	27.07	61.51	60.54	79.06	65.20	74.04	24.36	35.00	27.25
11	例-比例型、$K = 2^{18}$	81.67	46.85	91.63	91.99	88.73	87.68	87.20	86.93	90.35	83.30	84.01	91.47	73.29	40.96	19.07	38.43	78.17	86.74	67.94	76.57	78.88	87.50	62.00	67.70	30.85	63.43	62.54	76.53	65.67	67.31	24.57	35.19	27.39
11	例-比例型、$K = 2^{19}$	81.42	45.94	91.63	92.20	89.22	88.44	88.32	86.84	90.10	83.73	84.29	91.84	70.40	41.26	19.24	38.71	79.78	88.15	67.30	75.66	75.59	87.50	59.00	68.22	30.64	65.32	64.29	73.65	65.05	69.23	25.21	36.30	27.76
11	例-比例型、$K = 2^{20}$	80.80	42.55	92.78	91.27	87.99	88.36	88.10	86.10	89.62	84.15	84.26	92.20	68.95	41.05	19.24	38.46	80.36	88.27	67.38	73.21	76.18	83.93	62.00	67.57	26.86	66.12	65.22	76.90	64.73	69.23	25.66	36.93	27.68
11	例-比例型、$K = 2^{21}$	79.83	44.45	91.28	89.00	84.31	87.54	87.40	84.93	88.53	82.54	84.16	90.85	67.87	40.51	18.79	37.92	79.50	87.48	65.10	71.16	68.88	85.71	57.00	62.75	23.40	64.50	63.65	72.92	64.11	71.15	25.82	37.22	27.13
11	温度スケール、T = 2	81.90	54.00	91.74	90.56	86.76	85.11	84.60	86.40	89.74	83.47	84.15	91.51	72.56	41.09	19.28	38.54	79.42	87.77	69.92	76.73	92.37	92.86	57.00	69.80	31.90	66.65	65.74	72.92	67.08	75.96	25.42	36.72	27.20
11	温度スケーリング、T = 4	80.56	45.38	91.97	89.68	85.78	83.13	82.76	86.39	90.00	82.78	84.19	91.16	73.65	41.09	19.22	38.51	77.99	86.81	69.54	76.76	97.36	96.43	59.00	68.10	31.48	64.26	63.27	74.73	64.26	71.15	25.04	35.82	27.45
11	温度スケーリング、T = 8	77.21	40.07	91.06	88.11	83.33	79.20	79.06	86.60	89.90	83.05	83.56	90.21	59.93	41.01	19.10	38.40	77.14	85.99	66.07	73.94	93.70	94.64	60.00	66.36	26.86	63.46	62.60	62.09	63.32	65.38	24.55	35.35	27.17
12	F 教師なし事前学習 + ファインチューニング	83.28	53.84	92.68	92.07	88.92	88.02	87.94	88.67	91.56	84.24	84.57	90.48	76.28	41.33	19.24	38.77	80.88	88.81	71.36	76.62	91.22	91.96	66.20	66.13	25.78	69.05	68.16	75.34	68.04	78.56	26.98	39.82	27.65
12	マルチタスクトレーニング	81.42	45.94	91.63	92.20	89.22	88.44	88.32	86.84	90.10	83.73	84.29	91.84	70.40	41.26	19.24	38.71	79.78	88.15	67.30	75.66	75.59	87.50	59.00	68.22	30.64	65.32	64.29	73.65	65.05	69.23	25.21	36.30	27.76
12	マルチタスクの事前トレーニング + ファインチューニング	83.11	51.42	92.66	91.73	88.73	88.06	87.70	88.61	91.61	84.09	84.31	91.85	76.53	41.15	19.12	38.59	80.26	88.50	71.03	79.54	81.69	87.50	65.00	70.72	31.48	65.94	65.03	81.23	68.18	73.08	27.08	39.80	28.07
12	放置型マルチタスクトレーニング	81.98	48.00	93.23	91.72	88.24	87.76	87.32	88.61	91.44	84.00	84.11	90.79	72.20	41.34	19.05	38.77	79.97	88.10	71.68	78.35	86.76	89.29	66.00	68.09	29.49	66.23	65.27	79.06	68.65	78.85	26.93	39.79	27.87
12	教師ありのマルチタスク事前学習	79.93	36.60	92.43	91.58	88.24	87.03	86.78	88.15	91.20	82.87	83.16	90.13	70.76	41.12	18.96	38.49	77.38	85.65	65.36	75.66	68.87	83.93	58.00	64.81	21.93	55.37	54.61	71.12	67.40	75.96	26.81	40.13	28.04
13	F ベースライン	83.28	53.84	92.68	92.07	88.92	88.02	87.94	88.67	91.56	84.24	84.57	90.48	76.28	41.33	19.24	38.77	80.88	88.81	71.36	76.62	91.22	91.96	66.20	66.13	25.78	69.05	68.16	75.34	68.04	78.56	26.98	39.82	27.65
13	*1*×サイズ、*4*×トレーニングステップ	85.33	60.29	93.81	94.06	91.67	89.42	89.25	89.15	91.87	86.01	85.70	91.63	78.34	41.52	19.33	38.96	82.45	90.19	74.72	79.17	94.75	92.86	71.00	67.34	29.70	72.63	71.59	78.34	72.10	82.69	27.08	40.66	27.93
13	*1*倍サイズ、*4*倍バッチサイズ	84.60	56.08	93.12	92.31	89.22	88.85	88.84	89.35	92.07	85.98	86.13	91.07	80.14	41.70	19.42	39.08	82.52	90.21	74.64	78.78	93.69	94.64	72.00	68.09	30.95	74.73	73.90	76.53	70.06	81.73	27.07	40.60	27.84
13	*2*×サイズ、*2*×トレーニングステップ	86.18	62.04	93.69	93.36	90.69	89.18	89.23	89.35	92.05	87.23	87.05	92.68	81.95	41.74	19.66	39.14	84.18	91.29	77.18	80.98	97.36	96.43	74.00	71.34	35.68	77.11	76.34	80.51	69.28	85.58	27.52	41.03	28.19
13	*4*×サイズ、*1*×トレーニングステップ	85.91	57.58	94.38	92.67	89.95	89.60	89.60	89.44	92.14	87.05	87.12	93.12	83.39	41.60	19.73	39.08	83.86	91.32	78.04	81.38	89.09	94.64	73.00	73.74	40.40	78.25	77.40	81.59	70.22	91.35	27.47	40.71	28.10
13	*4*× アンサンブル	84.77	56.14	93.46	93.31	90.67	89.71	89.60	89.62	92.24	86.22	86.53	91.60	77.98	42.10	20.10	39.56	83.09	90.40	71.74	77.58	89.85	91.07	66.00	69.32	29.49	72.67	71.94	76.90	69.12	72.12	28.05	40.53	28.09
13	*4*× アンサンブル、ファインチューンのみ	84.05	54.78	92.78	93.15	90.44	88.34	88.12	89.27	91.97	85.33	85.88	90.98	77.62	41.66	19.57	39.12	82.36	89.86	71.56	77.43	90.07	92.86	69.00	67.31	26.34	70.47	69.64	75.45	68.18	74.04	27.55	40.22	28.09

表16：本論文のすべての実験について、検討したすべてのタスクで達成したスコア。最初の列には、ある実験について凝縮された結果が示されたテーブルを列挙する。本文と同様に、Ｆと書かれた行は、我々のベースラインモデル（3.1節で説明）を示す。

参考文献

Rami Al-Rfou, Dokook Choe, Noah Constant, Mandy Guo, and Llion Jones.より深い自己注意を伴う文字レベルの言語モデリング。*人工知能に関するAAAI会議の議事録*、2019年において。

ローハン・アニル、ヴィニート・グプタ、トマー・コレン、ヨーラム・シンガー。大規模学習のためのメモリ効率の良い適応的最適化. *arXiv preprint arXiv:1901.11150*, 2019.

Naveen Arivazhagan, Ankur Bapna, Orhan Firat, Dmitry Lepikhin, Melvin Johnson, Maxim Krikun, Mia Xu Chen, Yuan Cao, George Foster, Colin Cherry, et al. Massively multilingual neural machine translation in the wild： Findings and challenges. *arXiv preprint arXiv:1907.05019*, 2019.

ジミー・レイ・バ、ジェイミー・ライアン・キロス、ジェフリー・E・ヒントン。レイヤー正規化. *arXiv preprint arXiv:1607.06450*, 2016.

アレクセイ・バエフスキー、セルゲイ・エドゥノフ、インハン・リュウ、ルーク・ゼトルモイヤー、マイケル・アウリ。Cloze- driven pre-training of self-attention networks. *arXiv preprint arXiv:1903.07785*, 2019.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio.整列と翻訳を共同で学習することによるニューラル機械翻訳．In *Third International Conference on Learning Representations*, 2015.

Ankur Bapna、Naveen Arivazhagan、Orhan Firat。ニューラル機械翻訳のためのシンプルでスケーラブルな適応。 *arXiv preprint arXiv:1909.08478*, 2019.

Iz Beltagy, Kyle Lo, and Arman Cohan.SciBERT: A pretrained language model for scientific text.In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2019.

Ondřej Bojar, Christian Buck, Christian Federmann, Barry Haddow, Philipp Koehn, Johannes Leveling, Christof Monz, Pavel Pecina, Matt Post, Herve Saint-Amand, et al. Findings of the 2014 workshop on statistical machine translation.統計的機械翻訳に関する*第9回ワークショップの議事録*, 2014.

Ondřej Bojar, Rajen Chatterjee, Christian Federmann, Barry Haddow, Matthias Huck, Chris Hokamp, Philipp Koehn, Varvara Logacheva, Christof Monz, Matteo Negri, et al. Findings of the 2015 workshop on statistical machine translation.In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, 2015.

Ondřej Bojar, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, Varvara Logacheva, Christof Monz, et al. Findings of the 2016 conference on machine translation.機械翻訳に関する*第1回会議の議事録*, 2016.

Samuel R. Bowman, Luke Vilnis, Oriol Vinyals, Andrew M. Dai, Rafal Jozefowicz, and Samy Bengio.連続空間からの文の生成. *arXiv preprint arXiv:1511.06349*, 2015.

Christian Buck, Kenneth Heafield, and Bas Van Ooyen.N-gram counts and language models from the common crawl.In *LREC*, 2014.

リッチ・カルアナマルチタスク学習.*機械学習*, 28(1), 1997.

ダニエル・サー、モナ・ディアブ、エネコ・アギレ、イニゴ・ロペス＝ガスピオ、ルシア・スペシア。Semeval-2017 task 1: Semantic textual similarity-multilingual and crosslingual focused evaluation. *arXiv preprint arXiv:1708.00055*, 2017.

Jianpeng Cheng, Li Dong, and Mirella Lapata.Long short-term memory-networks for machine reading. *arXiv preprint arXiv:1601.06733*, 2016.

クリストファー・クラーク、ケントン・リー、ミン・ウェイ・チャン、トム・クウィアトコウスキー、マイケル・コリンズ、クリスティナ・トウタノワ。BoolQ: Exploring the surprising difficulty of natural yes/no questions. *arXiv preprint arXiv:1905.10044*, 2019.

(1)は、(2)は、(3)は、(4)は、(5)です。Electra：テキストエンコーダを生成器ではなく識別器として事前学習させる。

アレクシス・コネオ、ドゥーエ・キエラ。SentEval: An evaluation toolkit for universal sentence representations. *arXiv preprint arXiv:1803.05449*, 2018.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loic Barrault, and Antoine Bordes.自然言語推論データからの普遍的文表現の超ビジョン学習. *arXiv preprint arXiv:1705.02364*, 2017.

Ido Dagan, Oren Glickman, and Bernardo Magnini.PASCAL recognising textual entailment challenge.*機械学習課題ワークショップ*, 2005.

アンドリュー M. ダイ、クオック V. レ。半教師付き配列学習。In *Advances in neural information processing systems*, 2015.

マリー＝カトリーヌ・デ・マルネフ、マンディ・シモンズ、ジュディス・トンハウザー。コミットメントバンク：自然発生する談話における投影を調査する。*Sinn und Bedeutung 23*, 2019 に掲載されています。

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei.ImageNet：大規模な階層型画像データベース。*2009 IEEE conference on computer vision and pattern*

recognition, 2009に掲載。

ジェイコブ・デブリン、ミン・ウェイ・チャン、ケントン・リー、クリスティナ・トゥータノヴァ。BERT: Pre- training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

ウィリアム・B・ドラン、クリス・ブロケット。文節パラフレーズのコーパスを自動的に構築する。*Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*, 2005.

Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. Unified language model pre-training for natural language understanding and generation. *arXiv preprint arXiv:1905.03197*, 2019.

セルゲイ・エドゥノフ、マイレ・オット、マイケル・アウリ、デイヴィッド・グランジェ。スケールで逆翻訳を理解する。 *arXiv preprint arXiv:1808.09381*, 2018.

Edouard Grave, Piotr Bojanowski, Prakhar Gupta, Armand Joulin, and Tomas Mikolov. 157言語の単語ベクトルを学習する。 *arXiv preprint arXiv:1802.06893*, 2018.

アレックス・グレイヴスリカレントニューラルネットワークでシーケンスを生成する。 *arXiv preprint arXiv:1308.0850*, 2013.

Ivan Habernal, Omnia Zayed, and Iryna Gurevych.C4Corpus：多言語ウェブサイズコーパスをフリーライセンスで提供。*Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 914-922, 2016.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun.画像認識のための深層残差学習． *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016.

Kaiming He 、 Ross Girshick 、 Piotr Dollár.Rethinking ImageNet pre-training. *arXiv preprint arXiv:1811.08883*, 2018.

Pengcheng He, Xiaodong Liu, Weizhu Chen, and Jianfeng Gao.A hybrid neural network model for commonsense reasoning. *arXiv preprint arXiv:1907.11983*, 2019.

カール・モーリッツ・ヘルマン、トマス・コシスキー、エドワード・グレフェンステット、ラッセ・エスペホルト、ウィル・ケイ、ムスタファ・スレイマン、フィル・ブルンソム。機械に読み解くことを教える。In *Advances in neural information processing systems*, 2015.

Joel Hestness, Sharan Narang, Newsha Ardalani, Gregory Diamos, Heewoo Jun, Hassan Kianinejad, Md.Mostofa Ali Patwary, Yang Yang, and Yanqi Zhou.ディープラーニングのスケーリングは予測可能、経験的に。 *arXiv preprint arXiv:1712.00409*, 2017.

フェリックス・ヒル、キョンヒョン・チョー、アンナ・コルホネン。ラベルのないデータから文の分散表現を学習する。 *arXiv preprint arXiv:1602.03483*, 2016.

ジェフリー・ヒントン、オリオール・ヴィンヤルス、ジェフ・ディーン。ニューラル

ネットワークの知識を抽出する。
arXiv preprint arXiv:1503.02531, 2015.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly.NLP のためのパラメータ効率の良い転移学習. *arXiv preprint arXiv:1902.00751*, 2019.

ジェレミー・ハワードとセバスチャン・ルーダー。Universal language model fine-tuning for text classifi-cation. *arXiv preprint arXiv:1801.06146*, 2018.

Cheng-Zhi Anna Huang, Ashish Vaswani, Jakob Uszkoreit, Ian Simon, Curtis Hawthorne, Noam Shazeer, Andrew M. Dai, Matthew D. Hoffman, Monica Dinculescu, and Dou- glas Eck.Music transformer： 長期的な構造を持つ音楽を生成する。 In *Seventh International Conference on Learning Representations*, 2018a.

黄延平、Yonglong Cheng、Dehao Chen、HyoukJoong Lee、Jiquan Ngiam、Quoc V Le、およびZhifeng Chen。GPipe：パイプライン並列性を用いた巨大ニューラルネットワークの効率的なトレーニング. *arXiv preprint arXiv:1811.06965*, 2018b.

Minyoung Huh, Pulkit Agrawal, and Alexei A. Efros.What makes ImageNet good for transfer learning? *arXiv preprint arXiv:1608.08614*, 2016.

Shankar Iyer, Nikhil Dandekar, and Kornel Csernai.最初のQuoraデータセットのリリース：Question pairs. https://data.quora.com/First-Quora-Dataset-Release-Question-Pairs, 2017.

Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell.Caffe：高速な特徴埋込のための畳み込みアーキテクチャ。*Proceedings of the 22nd ACM international conference on Multimedia*, 2014.

Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu.TinyBERT: Distilling BERT for natural language understanding. *arXiv preprint arXiv:1909.10351*, 2019.

マンダル・ジョシ、チェ・ウンソル、ダニエル・S・ウェルド、ルーク・ゼットルモイヤー。TriviaQA： A large scale distantly supervised challenge dataset for reading comprehension. *arXiv preprint arXiv:1705.03551*, 2017.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levie.SpanBERT: Improving pre-training by representing and predicting spans. *arXiv preprint arXiv:1907.10529*, 2019.

Rafal Jozefowicz, Oriol Vinyals, Mike Schuster, Noam Shazeer, and Yonghui Wu.言語モデリングの限界を探る。 *arXiv preprint arXiv:1602.02410*, 2016.

ナル・カルチブレナー、エドワード・グレフェンステット、フィル・ブルンソム。文章をモデリングするための畳み込みニューラルネットワーク。*計算言語学会の第52回年次大会*講演論文集, 2014.

Nitish Shirish Keskar, Bryan McCann, Lav R. Varshney, Caiming Xiong, and Richard Socher.CTRL：制御可能な生成のための条件付き変換言語モデル. *arXiv preprint arXiv:1909.05858*, 2019a.

ニティッシュ・シリシュ・ケスカー、ブライアン・マッカン、カイミン・シオン、リチャード・ソッチャー。スパン抽出による質問応答とテキスト分類の統一. *arXiv preprint arXiv:1904.09286*, 2019b.

ダニエル・カシャビ、スニグダ・チャトルヴェディ、マイケル・ロス、シャム・ウパディヤイ、ダン・ロス。表面の向こう側を見る：複数の文に対する読解のためのチャレンジセット。*Proceedings of North American Chapter of the Association for Computational Linguistics (NAACL)*, 2018.

Ryan Kiros, Yukun Zhu, Ruslan R. Salakhutdinov, Richard Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler.スキップソートベクターIn *Advances in neural information processing systems*, 2015.

Vid Kocijan、Ana-Maria Cretu、Oana-Maria Camburu、Yordan Yordanov、およびThomas Lukasiewicz。Winograd schema challengeに対する驚くほど堅牢なトリック。*arXiv preprint arXiv:1905.06290*, 2019.

Jakub Konečny`, Brendan McMahan, and Daniel Ramage.Federated optimization：データセンターを超える分散最適化. *arXiv preprint arXiv:1511.03575*, 2015.

Jakub Konečny`, H. Brendan McMahan, Felix X. Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon.Federated learning：コミュニケーション効率向上のための戦略. *arXiv preprint arXiv:1610.05492*, 2016.

サイモン・コーンブリス、ジョナサン・シュレンズ、クオック・V・レ。より良いImageNetモデルはより良い転送を行うか？ *arXiv preprint arXiv:1805.08974*, 2018.

アレックス・クリシェフスキー畳み込みニューラルネットワークを並列化するための1つの奇妙なトリック。 *arXiv preprint arXiv:1404.5997*, 2014.

工藤拓。サブワード正則化：複数のサブワード候補を持つニューラルネットワーク翻訳モデルの改善. *arXiv preprint arXiv:1804.10959*, 2018.

工藤拓とジョン・リチャードソン。SentencePiece：A simple and language independent sub word tokenizer and detokenizer for neural text processing. *arXiv preprint arXiv:1808.06226*, 2018.

ギョーム・ランプル、アレクシス・コンノー。クロスリンガル言語モデルのプリトレーニング. *arXiv preprint arXiv:1901.07291*, 2019.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut.ALBERT: A lite BERT for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*, 2019.

ヘクター・ルベスク、アーネスト・デイビス、レオラ・モーゲンシュテルン。Winograd schema challenge.In *Thirteenth International Conference on the Principles of Knowledge Representation and Reasoning*, 2012.

Qi Li.文献調査：自然言語処理のためのドメイン適応アルゴリズム.
2012.

チン・ユー・リンROUGE：要約の自動評価のためのパッケージ。In *Text summarization branches out*, 2004.

ピーター・J・リュー、モハマド・サレハ、エティエンヌ・ポット、ベン・グッドリッチ、ライアン・セパシー、ルカシュ・カイザー、ノーム・シャゼア。長い配列を要約することでウィキペディアを生成する. *arXiv preprint arXiv:1801.10198*, 2018.

ピーター・J・リュウ、ユアン・チョン、ジー・レン。SummAE: Zero-shot abstractive text summarization using length-agnostic auto-encoders. *arXiv preprint arXiv:1910.00998*, 2019a.

Xiaodong Liu, Jianfeng Gao, Xiaodong He, Li Deng, Kevin Duh, and Ye-Yi Wang.意味分類と情報検索のためのマルチタスク深層ニューラルネットワークを用いたレペゼンセーション学習.*Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics：Human Language Technologies*, 2015.

Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao.自然言語理解のためのマルチタスク深層ニューラルネットワーク. *arXiv preprint arXiv:1901.11504*, 2019b.

ヤン・リウ。抽出的要約のためのBERTを微調整する。 *arXiv preprint arXiv:1903.10318*, 2019.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov.RoBERTa：ロバストな最適化されたBERTプレトレーニングアプローチ。 *arXiv preprint arXiv:1907.11692*, 2019c.

ラジャヌゲン・ロゲスワラン、ホンラック・リー。文の表現を学習するための効率的なフレームワーク. *arXiv preprint arXiv:1803.02893*, 2018.

Dhruv Mahajan, Ross Girshick, Vignesh Ramanathan, Kaiming He, Manohar Paluri, Yixuan Li, Ashwin Bharambe, and Laurens van der Maaten.弱教師付きプリトレーニングの限界を探る。In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.

ブライアン・マッキャン、ニティッシュ・シリシュ・ケスカー、カイミン・シオン、リチャード・ソッチャー。The nat- ural language decathlon：質問応答としてのマルチタスク学習. *arXiv preprint arXiv:1806.08730*, 2018.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean.ベクトル空間における単語表現の効率的な推定.*arXiv preprint arXiv:1301.3781*, 2013a.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeff Dean.単語とフレーズの分散表現とその構成性。In *Advances in neural information processing systems*, 2013b.

Ramesh Nallapati、Bowen Zhou、Cicero Nogueira dos santos、Caglar Gulcehre、および

Bing Xiang。sequence-to-sequence RNNs and beyondを用いた抽象的なテキスト要約. *arXiv preprint arXiv:1602.06023*, 2016.

マキシム・オカブ、レオン・ボトゥー、イヴァン・ラプテフ、ジョセフ・シビック。畳み込みニューラルネットワークを用いた中レベル画像表現の学習と転送。*Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu.BLEU：機械翻訳の自動評価のための方法。*計算言語学会の第40回年次大会予稿集*。計算言語学協会, 2002.

ロマン・パウルス、カイミン・シオン、リチャード・ソッチャー。抽象的な要約のための深い強化モデル. *arXiv preprint arXiv:1705.04304*, 2017.

ジェフリー・ペニントン、リチャード・ソッチャー、クリストファー・マニング。GloVe：単語表現のためのグローバルベクター。*Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014.

マシュー・ピーターズ、セバスチャン・ルーダー、ノア・A・スミス。To tune or not to tune? プリトレーニングされた表現を多様なタスクに適応させる*arXiv preprint arXiv:1903.05987*, 2019.

マシュー・E・ピーターズ、マーク・ノイマン、モヒト・アイヤー、マット・ガードナー、クリストファー・クラーク、ケントン・リー、ルーク・ゼットルモイヤー。深い文脈に基づく単語表現. *arXiv preprint arXiv:1802.05365*, 2018.

Jason Phang, Thibault Févry, and Samuel R. Bowman.STILT上のセンテンスエンコーダ：中間的なラベル付きデータタスクでの上乗せ学習. *arXiv preprint arXiv:1811.01088*, 2018.

Mohammad Taher Pilehvar、 Jose Camacho-Collados.WIC: 10,000 example pairs for evaluating context-sensitive representations. *arXiv preprint arXiv:1808.09121*, 2018.

マット・ポストBLEUスコアの報告における明確化の呼びかけ. *arXiv preprint arXiv:1804.08771*, 2018.

アレック・ラドフォード、カーティク・ナラシマン、ティム・サリマンス、イリヤ・スーツケバー。Generative pre-trainingによる言語理解の向上, 2018.

アレック・ラドフォード、ジェフリー・ウー、リウォン・チャイルド、デヴィッド・ルアン、ダリオ・アモデイ、イリヤ・スツキーヴァー。
言語モデルは教師なしマルチタスク学習者 2019年版

Altaf RahmanとVincent Ng。定冠詞の複雑なケースの解決：Winogradスキーマの挑戦.*自然言語処理における経験的方法と計算された自然言語学習に関する2012年合同会議議事*録.計算言語学協会, 2012.

プラナフ・ラージプルカー、ジャン・ジャン、コンスタンチン・ロピレフ、パーシー・リャンスクワッド：テキストの機械理解のための100,000以上の質問。 *arXiv preprint arXiv:1606.05250*, 2016.

プラジット・ラマチャンドラン、ピーター・J・リュー、クオック・V・レ。Unsupervised pre-training for sequence to sequence learning. *arXiv preprint arXiv:1611.02683*, 2016.

Alex Ratner, Braden Hancock, Jared Dunnmon, Roger Goldman, and Christopher Ré.Snorkel MeTaL: Weak supervision for multi-task learning.In *Proceedings of the Second Workshop on Data Management for End-To-End Machine Learning*, 2018.

メリッサ・ローメイル、コスミン・アドリアン・ベジャン、アンドリュー・S・ゴードン。もっともらしい選択肢の選択：コモンセンス因果推論の評価.In *2011 AAAI Spring Symposium Series*, 2011.

セバスチャン・ルーダー（Sebastian Ruder）。ディープニューラルネットワークにおけるマルチタスク学習の概要. *arXiv preprint arXiv:1706.05098*, 2017.

セバスチャン・ルーダー *自然言語処理のためのニューラル・トランスファー・ラーニング(Neural Transfer Learning for Natural Language Processing)*.博士論文、NUIゴールウェイ、2019年。

セバスチャン・ルーダー、マシュー・E・ピータース、スワバ・スワヤムディプタ、トーマス・ウルフ。自然言語処理における転移学習。*Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics（計算言語学会の北米支部の2019年会議）*において：*Tutorials*, pages 15-18, 2019.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. ImageNet large scale visual recognition challenge.*International journal of computer vision*, 2015.

ヴィクター・サン、ライサンドル・ドゥブット、ジュリアン・ショーモン、トーマス・ウルフ。DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2019.

アビゲイル・シー、ピーター・J・リュー、クリストファー・D・マニング。Get to the point：ポインタジェネレータネットワークによる要約. *arXiv preprint arXiv:1704.04368*, 2017.

リコ・センリッチ、バリー・ハドウ、アレクサンドラ・バーチ。サブワード単位による希少語のニューラル機械翻訳. *arXiv preprint arXiv:1508.07909*, 2015.

Christopher J Shallue, Jaehoon Lee, Joe Antognini, Jascha Sohl-Dickstein, Roy Frostig, and George E. Dahl.ニューラルネットワークのトレーニングにおけるデータ並列性の効果を測定する。 *arXiv preprint arXiv:1811.03600*, 2018.

ピーター・ショウ、ヤコブ・ウスコレイト、アシシュ・ヴァスワニ。相対位置表現による自己アテンション. *arXiv preprint arXiv:1803.02155*, 2018.

ノーム・シャゼール、ミッチェル・スターンAdafactor：サブリニアメモリコストによる適応的な学習率. *arXiv preprint arXiv:1804.04235*, 2018.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean.とんでもなく大きなニューラルネットワーク： The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538*, 2017.

Noam Shazeer, Youlong Cheng, Niki Parmar, Dustin Tran, Ashish Vaswani, Penporn Koanantakool, Peter Hawkins, HyoukJoong Lee, Mingsheng Hong, Cliff Young, Ryan Sepassi, and Blake Hechtman.Mesh-tensorflow：スーパーコンピュータのための

ディープラーニング。In *Advances in Neural Information Processing Systems*, 2018.

Jason R. Smith, Herve Saint-Amand, Magdalena Plamada, Philipp Koehn, Chris Callison-Burch, and Adam Lopez.一般的なクロールからの安価なウェブスケールパラレルテキスト。*計算言語学会の第51回年次大会*（2013年）で発表されました。

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts.Recursive deep models for semantic compositionality over a sentiment treebank.*自然言語処理における経験的方法に関する2013年会議の議事録*, 2013.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu.MASS: Masked sequence to sequence pre-training for language generation. *arXiv preprint arXiv:1905.02450*, 2019.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdii- nov.Dropout: a simple way to prevent neural networks from overfitting. *機械学習研究ジャーナル*, 2014.

サンディープ・スブラマニアン、アダム・トリシュラー、ヨシュア・ベンジオ、クリストファー・J・パル。大規模マルチタスク学習による汎用分散文表現の学習. *arXiv preprint arXiv:1804.00079*, 2018.

イリヤ・ソーツケバー、オリオル・ビニャルズ、クオック・V・レ。ニューラルネットを用いた配列間学習。In *Advances in neural information processing systems*, 2014.

リチャード・S・サットン The bitter lesson. http://www.incompleteideas.net/IncIdeas/ BitterLesson.html, 2019.

ウィルソン・L・テイラー"Cloze手続き"：読みやすさを測る新しいツール。*Journalism Bulletin*, 1953.

トリウ・H．トリン、クオック・V・レ。コモンセンス推論のためのシンプルな方法. *arXiv preprint arXiv:1806.02847*, 2018.

Adam Trischler、Tong Wang、Xingdi Yuan、Justin Harris、Alessandro Sordoni、Philip Bachman、Kaheer Suleman。NewsQA：機械理解データセット. *arXiv preprint arXiv:1611.09830*, 2016.

アシシュ・ヴァスワニ、ノーム・シャゼール、ニキ・パーマー、ヤコブ・ウスコレイト、リオン・ジョーンズ、エイダン・N・ゴメス、ウカシュ・カイザー、イリア・ポロスヒンAttention is all you need.In *Advances in neural information processing systems*, 2017.

エレナ・ヴォイタ、リコ・センリッチ、イワン・ティトフ。トランスフォーマーにおける表現のボトムアップ進化：機械翻訳と言語モデリングを目的とした研究。*arXiv preprint arXiv:1909.01380*, 2019.

Alex Wang, Amapreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman.GLUE: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*, 2018.

Alex Wang, Jan Hula, Patrick Xia, Raghavendra Pappagari, R. Thomas McCoy, Roma Patel, Najoung Kim, Ian Tenney, Yinghui Huang, Katherin Yu, et al. Can you tell me how to get

past Sesame Street? 言語モデリングを超えた文レベルのプリトレーニング。 *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019a.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman.SuperGLUE: A stickier benchmark for general-purpose language understanding systems. *arXiv preprint arXiv:1905.00537*, 2019b.

Wei Wang, Bin Bi, Ming Yan, Chen Wu, Zuyi Bao, Liwei Peng, and Luo Si.StructBERT: Incorporating language structures into pre-training for deep language understanding. *arXiv preprint arXiv:1908.04577*, 2019c.

アレックス・ウォースタット、アマンプリート・シン、サミュエル・R・ボウマン。ニューラルネットワークの受容性判断. *arXiv preprint arXiv:1805.12471*, 2018.

アディナ・ウィリアムズ、ニキータ・ナンギア、サミュエル・R・ボウマン。推論による文章理解のための広範なカバレッジのチャレンジコーパス. *arXiv preprint arXiv:1704.05426*, 2017.

ロナルド・J・ウィリアムズ、デビッド・ジプサー。完全リカレント型ニューラルネットワークの継続的な実行のための学習アルゴリズム。*ニューラル・コンピュテーション*, 1989.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. Googleのニューラル機械翻訳システム：人間と機械翻訳のギャップを埋める。 *arXiv preprint arXiv:1609.08144*, 2016.

楊志林、戴志芳、楊依明、Jaime Carbonell、Ruslan Salakhutdinov、Quoc V. Le.XLNet：言語理解のための一般化された自己回帰的な事前学習. *arXiv preprint arXiv:1906.08237*, 2019.

ジェイソン・ヨシンスキー、ジェフ・クルーン、ヨシュア・ベンジオ、ホド・リプソン。ディープニューラルネットワークにおける特徴はどの程度移植可能か？In *Advances in neural information processing systems*, 2014.

Adams Wei Yu, David Dohan, Minh-Thang Luong, Rui Zhao, Kai Chen, Mohammad Norouzi, and Quoc V. Le.QAnet：読解のためのローカルコンボリューションとグローバルセルフアテンションの組み合わせ. *arXiv preprint arXiv:1804.09541*, 2018.

ローワン・ゼラーズ、アリ・ホルツマン、ハンナ・ラシュキン、ヨナタン・ビスク、アリ・ファルハディ、フランツィスカ・ロエスナー、チェ・ヨジン。ニューラルフェイクニュースに対する防御. *arXiv preprint arXiv:1905.12616*, 2019.

Sheng Zhang, Xiaodong Liu, Jingjing Liu, Jianfeng Gao, Kevin Duh, and Benjamin Van Durme.ReCoRD: Bridging the gap between human and machine commonsense reading comprehension. *arXiv preprint arXiv:1810.12885*, 2018.

Chen Zhu, Yu Cheng, Zhe Gan, Siqi Sun, Thomas Goldstein, and Jingjing Liu.Freelb：En-

hanced adversarial training for language understanding. *arXiv preprint arXiv:1909.11764*, 2019.

朱裕勲、ライアン・キロス、リッチ・ゼメル、ルスラン・サラクートディノフ、ラクエル・ウルタスン、アントニオ・トラルバ、サンヤ・フィドラー。本と映画の整合性：映画を見たり本を読んだりすることで、ストーリーのような視覚的な説明の国家を目指す。In *Proceedings of the IEEE international conference on computer vision*, 2015.