豊田工業大学大学院　修士論文

文献からのグラフ構造抽出のための
ニューラル編集モデル

2021年2月

工学研究科　先端工学専攻
知能数理研究室
19446　　牧野 晃平

目次

図目次

表目次

1. 序論

1.1 背景

文献中に記述された情報を抽出して，利用する研究が盛んに行われている [5–7]．論文や Web ページのような文献中に記述された情報は，人間が普段の生活で使用する自然言語によって表現されており，構造化されていないため，情報を利用するためには人間が文献を読んで解釈する必要がある．しかしながら，人が読める文献の数や一度に把握可能な情報量は限られており，大規模な文献を調査するのは困難であるため，コンピュータによって機械的に自然言語で記述された情報を処理する自然言語処理技術が求められている．文献中の情報を利用するためには，文献中に記述された情報を構造化する情報抽出技術が必要である．

文献中からの抽出対象の情報は，必要な語句の集合や表形式，グラフ形式といったような様々な形式で表現可能で，その抽出も様々な形式の情報に適した抽出手法が必要である．既存のタスクの抽出対象は，文献中に記述されている人名や場所といったような固有表現を抽出する固有表現抽出，その固有表現同士の間の関係性を抽出する関係抽出 [8–10] や，表形式の情報抽出 [11]，出来事を抽出するイベント抽出 [12]，合成手順コーパスのようなグラフ形式の情報の抽出 [1] といったような形式があり，それぞれに対して特化した抽出手法が提案されている [5, 13, 14]．

このように，様々な形式の情報抽出を行うために，それぞれに特化した情報抽出器を利用することで，高い性能での抽出が試みられている．しかしながら，それぞれに対して特化したモデルを作成するのはコストが高いという問題点がある．それぞれの形式に特化したモデルを作成するためには，その形式に沿った手法を開発する必要がある．深層学習を用いた手法では，事前学習モデルを用いて出力部のみを変更することでモデルを作成する方法 [15] も考えられるが，特化したモデルと比較すると性能が不十分な場合がある [16]．

情報を表現する形式の一つとして，グラフ構造がある．グラフ構造は節点と，節点同士を関連性に基づいて結びつける辺の集合で表現できる形式であり，様々な情報を包括的に表現可能な形式

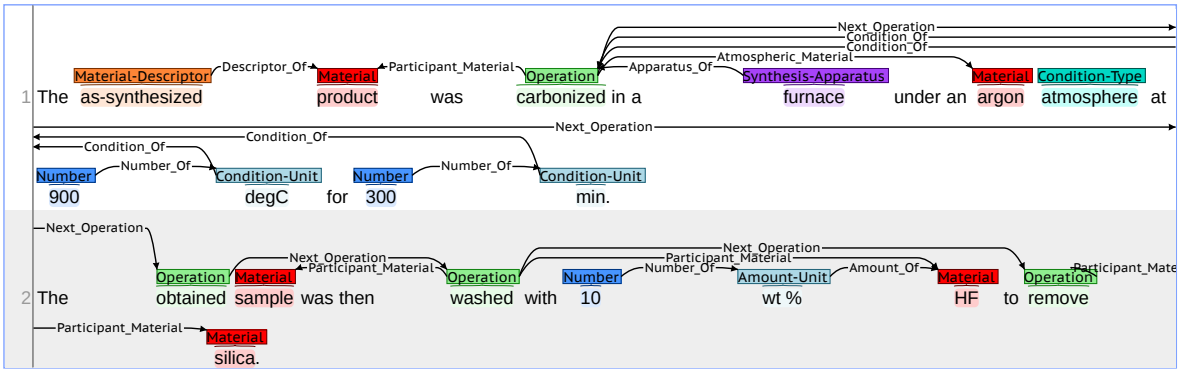


図1: 合成手順コーパス [1] における文章の例 [2]

の一つである．例えば，用語とその用語間の関係を抽出する関係抽出では，用語を節点，用語ペア間の関係を辺として定義することでグラフ構造として表現可能で，表形式は表の見出しとセルを節点として，見出しとその見出しに対する情報を辺で結びつけることで表現可能である．グラフ形式の情報を抽出可能な手法では，グラフ形式の情報に落とし込むことで，一つの手法で色々な形式の情報を抽出することが可能になる．

材料分野では，論文中に記述された材料の合成プロセスなどの情報を抽出し，検索システムや新規材料探索に役立てて，開発コストを削減する研究が注目されている [1, 11, 17, 18]．中でも無機材料分野では，新規材料を合成するための画一的な理論がなく，合成に必要な知識が人間の知識依存になるため，論文中から情報を抽出して構造化することで，検索や新規材料探索に利用することによって開発コストを低減する技術が期待されている [1, 11, 18]．無機合成のためには専門家によって条件を定める必要があり，例えば熱する温度や時間，混合する際の溶媒といったような条件は人手で経験に基づいて定める必要がある．合成プロセス情報は，プロセスの操作や材料などの情報を節点として，操作の進行や操作への材料の投入といった辺を定義することで，グラフ構造として定義可能である．

本研究では，グラフ構造の抽出に取り組み，高い需要がある無機合成における合成プロセスを対象にグラフの抽出を行う．無機合成の解析のために様々なコーパスが提案されていて，Mysore ら [1] は無機材料文献に対して材料や操作の扱いや数値条件などをフローグラフとして合成プロセスをタグ付けした合成手順コーパスを提案し，Kuniyoshi ら [18] は無機材料文献の中でも全固

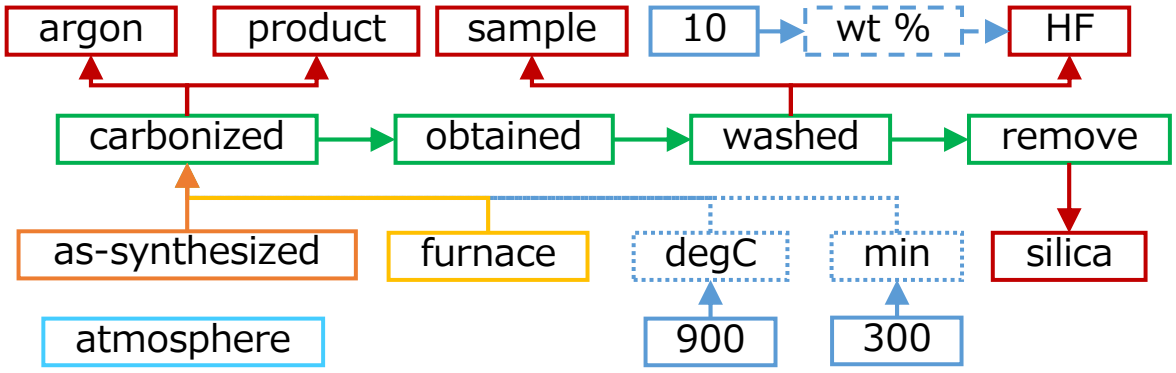


図2: 図 1から抽出した正解のグラフ

体電池に関する文献内の合成プロセスに対してタグ付けした．図 1は合成手順コーパスの一部を抜き出したもので，フローグラフとしてタグ付けされており，これをグラフとして表現した合成プロセスが図 2である．このグラフの抽出のためには節点となる用語の抽出と，その節点間の条件付けや材料の投入，プロセスの進行などの関係を表す辺の抽出が必要だが，本研究ではまずグラフの辺の抽出に取り組む．このようなフローグラフとして記述されている合成プロセスの辺抽出では，操作の進行のような内容は複数文に渡って記述されることが多く，文をまたいで記述されている用語同士を結びつける必要がある．

このプロセスのグラフの辺を抽出するのに似たタスクとして，文献中に記述された用語ペア間の関係を抽出する関係抽出タスクがある．関係抽出手法を利用して辺抽出するためには文をまたいだ語同士の関係抽出である，文間の関係抽出手法が必要となる．しかし，既存の用語間の関係抽出手法では，一文中に記述された用語同士の関係抽出手法 [5–7] がほとんどであり，文をまたぐ関係の関係抽出手法の提案は少ない [19]．また，文をまたいだ関係の抽出では，詳細なタグ付けが難しいことから，ほとんどが機械的にタグ付けされたデータに対する抽出の遠距離教師あり学習に対するタスク [20,21] に対する手法しか提案されておらず，教師あり学習のための手法は確立されていない．合成プロセスの抽出の場合，全体的な整合性から行われる操作の順番などを定める必要があり，例えば，温度などの条件はその操作に用いられる材料とその他に行われる操作，完成する生成物を考慮した上で定める必要があるため，周囲の関係も考慮して辺を抽出しなければならない．しかしながら，多くの文間の関係抽出手法 [6,19] では，部分的に用語ペア同士の関係のみ

に着目して辺となる関係を抽出しており，抽出される情報全体を考慮できていない．

1.2 目的

本研究では抽出する情報をグラフ構造として扱うことで，グラフ構造の情報抽出タスクに利用可能な抽出器を提案し，広く一般的に適用可能かつ高い性能での情報抽出を可能にすることを目的とする．グラフ構造は様々な情報を表現可能な構造で，例えば表のような構造ではそれぞれのセルを節点として扱い，その節点同士を表の情報を損なわないように辺でつなぐことで表現可能である．グラフ構造の情報抽出器を作成し，様々な情報抽出タスクの抽出対象をグラフ構造として扱うことで，様々なタスクに対して特化したモデルを利用せずに，一つの手法での抽出を可能にする．

情報をグラフとして扱うことで，グラフの視点での表現が獲得できる．グラフの視点で導入した表現は，グラフとしての整合性を保つために利用可能である．例えば合成プロセスの抽出では，一般的な用語間の関係を抽出するのとは異なり，それまでに投入された材料や今までの操作に基づいて，その次のプロセスが定まる．合成プロセスを単純な節点と辺の集合として抽出するのではなく，一続きのグラフとして抽出することで，プロセスとして整合性のある抽出を可能にする．

グラフの抽出は既存のグラフを事後編集して正しいグラフへと近づけるようにすることで行う．実際に企業などが利用する技術として，導入コストが低い技術が求められている．深層学習技術が発展したのは近年であることから，実際に導入するためには，既存のシステムを大幅に改修する必要があり，その改修コストが大きいと費用対効果が悪くなり導入する利点が小さくなるという問題点がある．そこで，既存のグラフを事後編集するモデルとすることで，既存のシステムで抽出されている情報をグラフに変換し，グラフを正しくなるように編集することで，深層学習を導入した性能が高い抽出を導入できる．さらに，既存の抽出されたグラフがない状態を初期状態として，既存のシステムを利用しない抽出も可能であることを示す．

本研究ではまず，節点については正解の情報を利用して，辺の抽出に取り組む．グラフの抽出を編集によって実現することから，辺と節点の編集が必要となる．しかしながら，節点の編集を行う

場合では，節点が変更されるため，辺の扱いが難しい．そこで，本研究ではまず編集が容易な辺を対象に取り組む．

情報抽出タスクであることを考慮し，文脈上とグラフ上の整合性がとれた抽出が可能なモデルを提案し，更に利用性を高めるために，既存のグラフを事後編集するモデルでこれを実現する．そのための手法として，人手で作成されたルールを利用するルールベースの抽出器で抽出されたグラフを深層学習モデルによって編集することで，文脈とグラフ全体での整合性がとれたグラフを抽出する．モデルでは文脈を考慮するために，用語の表現に大規模文献から言語の一般的な特徴を学習した事前学習モデル [3, 22] の出力を利用し，その表現をルールベースの抽出器から得られたグラフの節点に配置し，グラフ畳み込みネットワーク（GCN）[23] で辺に沿って情報を伝播させることでグラフの整合性を考慮する．

1.3 貢献

本研究の貢献は以下の点である．

- 文献からの情報抽出をグラフの抽出として定式化し，文脈とグラフの両者の観点での整合性をとる試みを行った．グラフの抽出として定式化することで，グラフとして表現可能な様々な情報抽出タスクに適用可能なモデルを提案した．文脈とグラフの両者の観点で整合性をとるために，文脈の観点では Transformer ベースの事前学習モデルを利用して得られる文脈情報を含んだ用語の表現を各節点に配置し，GCN によって辺に沿って節点の情報を伝搬させることで，文脈情報に加えてグラフの構造情報を明示的に付加したベクトルから辺の分類を行うモデルを利用した．
- グラフの編集モデルによって，既存のグラフを編集する手法を提案した．既存のグラフとして，辺が全く接続されていない節点のみのグラフを編集してグラフを抽出した場合と，ルールによって抽出したグラフを入力して編集した場合を比較して，ルールによって抽出されたグラフを編集したほうが高い性能での抽出が可能であることを示した．一方で，辺が

ない節点のみのグラフを編集して辺を徐々につける場合でも，一定の性能が得られることを示した．

- 合成プロセス抽出におけるグラフの抽出において，簡単な問題から難しい問題へと順番に解くことで，性能が向上することを示した．グラフの編集モデルでは，辺を編集する操作を順番に行い，その順序を文章中で隣接している節点同士から順番に，徐々に離れている節点同士の辺を編集するようにすることで，係り受けのような関係がわかりやすい簡単な問題から，徐々に離れて関係性がわかりにくい難しい問題へと解き進めるようにした．この操作によって，性能が向上することを示した．
- グラフの抽出器を利用して，合成プロセス抽出に取り組み，最先端の結果が得られた．需要の高い情報である無機材料文献における合成プロセスをまとめたコーパスである合成手順コーパスに対する抽出で，ルールで抽出したグラフと比較して，マクロ F 値において 0.1 ポイント程度の向上が見られ，最先端の結果が得られた．

1.4 構成

本論文の構成は，本章を含めて全 5 章からなる．2 章では本研究に関連する研究について述べる．そして，3 章では提案手法について述べる．提案手法はグラフを文脈上とグラフ上での整合性を考慮した深層学習モデルによる辺の分類器を利用して編集するモデルである．4 章では，提案手法によってルールベースの抽出器の出力を編集するモデルについて評価し，5 章ではまとめと，今後の課題について述べる．

2. 関連研究

本章では関連する研究について述べる．2.1 節では構造化された情報が付加された文献集合であるコーパスの内，本研究で抽出の対象とする合成プロセスの文献と関連するコーパスについて述べる．2.3 節では類似の手法である，用語とその間の関係を抽出する関係抽出手法について述べる．2.4 節では深層学習モデルとそのもととなるニューラルネットワークについて述べる．2.5 節では，ニューラルネットワーク構造の一つである Transformer [24] と，Transformer を利用して言語の基本的な情報を学習した事前学習モデルについて述べる．2.6 節ではグラフの特徴を得るためのニューラルネットワーク構造のグラフ畳み込みネットワークについて述べる．

2.1 合成手順コーパス

本研究では，Mysore らが作成・公開した，材料の作成手順である合成プロセスがタグ付けされている合成手順コーパス [1] からの合成プロセス抽出を対象とする．合成プロセスは材料の合成手順を示しており，図 1のように，材料や操作などの合成プロセスに関する用語を節点・材料の投入といったような用語同士の関係性を辺として，文献上にグラフが直接タグ付けされている．合成手順コーパスでは，用語ラベルは表 1に，辺ラベルとその辺ラベルが取りうる始点（Head）と終点（Tail）を表 2に示し，各辺の役割を表 3に示した．以降，ある Head と Tail 間の辺を（Head の用語ラベル）–（Tail の用語ラベル）として表現することとする．Operation–Material 間の辺については，イベント [25,26] としてタグ付けされているが，変換することでグラフとして扱うことが可能である．

無機材料分野では，材料合成のための確実な理論がなく，熟練した技術者が操作の方法や温度条件と言ったような条件を定める必要があり，開発コストが高くなってしまう．そこで，データ駆動の手法によって，既存の文献から材料を合成する手順である合成プロセスを抽出して利用することで，新規材料探索や検索システムに利用する動きがある [1,11,18]．Kononova ら [11] は固体

合成の合成プロセスをまとめ，大規模なコーパスを作成した．Kuniyoshi ら [18] は無機材料の中でも全固体電池に関する文献中の合成プロセスに限定したコーパスを作成し，その抽出に取り組んだ．このように，無機材料の合成プロセス情報は需要が高いことから，本研究の抽出対象は合成プロセスがタグ付けされたコーパスである合成手順コーパスを対象に抽出を行う．

文章で記述された手順を構造化することで利用性を高める必要があり，情報抽出技術によって構造化することが求められている．人間が何かを行うための行動や操作などを順序建ててまとめたものを手順といい，手順は様々な場面で文章化される．例えば，料理の手順では構造化された手順と文章が結び付けられたコーパス [27] だけでなく，kurashiru[*1]や cookpad[*2]と言ったようなウ

[*1] https://www.kurashiru.com/
[*2] https://cookpad.com/

表1: 合成手順コーパス [1] の用語ラベル

用語ラベル	説明
Operation	操作
Material	材料
Property-Misc	材料に関するその他の条件
Nonrecipe-Material	プロセスに関連しない材料
Property-Unit	材料に関する条件となる単位
Number	数値
Amount-Unit	材料の量に関する単位
Condition-Unit	操作に対する条件となる単位
Condition-Misc	操作に対するその他の条件
Amount-Misc	材料の量を示す条件
Material-Descriptor	材料に関する形状などの条件
Apparatus-Descriptor	装置に関する条件
Synthesis-Apparatus	プロセスに利用する装置
Characterization-Apparatus	顕微鏡などの評価のための装置
Property-Type	材料に関する数値条件に対する条件
Condition-Type	操作に関する数値条件に対する条件
Apparatus-Property-Type	装置に関する数値条件に対する条件
Apparatus-Unit	装置に対する数値条件の単位
Brand	材料や装置のブランド

ェブページにまとめられた文章もある．このような文章で記述された料理のレシピを解釈し，ロボットで自動的に調理する研究 [28] も行われており，文献として記述されている手順情報の構造化が要求されている．他にも湿式化学においては，実験手順をまとめたコーパス [29] が提案されており，実験を自動化する研究も行われている [30]．このように，文章で記述された手順を構造化する必要があり，そのための情報抽出技術が求められている．

表2: 合成手順コーパス [1] の用語ペアごとの辺ラベル

Head	Tail	辺ラベル
Operation	Operation	Next_Operation
Operation	Material	Recipe_Precursor Recipe_Target Participant_Material Solvent_Material Atmospheric_Material
Property-Misc	Material Nonrecipe-Material	Property_Of
Property-Unit	Material	
Condition-Unit Condition-Misc	Operation	Condition_Of
Number	Amount-Unit Condition-Unit	Number_Of
Amount-Unit	Material Nonrecipe-Material	Amount_Of
Amount-Misc	Material Nonrecipe-Material	
Material-Descriptor	Material Nonrecipe-Material	Descriptor_Of
Apparatus-Descriptor	Synthesis-Apparatus Characterization-Apparatus	
Property-Type Condition-Type Apparatus-Property-Type	Property-Unit Condition-Unit Apparatus-Unit	Type_Of
Brand	Material Nonrecipe-Material Synthesis-Apparatus Characterization-Apparatus	Brand_Of
Synthesis-Apparatus Characterization-Apparatus	Operation	Apparatus_Of
Apparatus-Unit	Synthesis-Apparatus Characterization-Apparatus	Apparatus_Attr_Of

手順の抽出のためには，手順全体での整合性をとる必要があり，グラフ構造とした場合ではグラフ全体で整合性が保たれている必要がある．例えば料理の手順の抽出を考えた時，灰汁を取る操作はそれまでの手順に灰汁が出るような食品が利用されている必要があったり，10 分で火を通すためには人参を切る大きさはある程度決まっていたりといったような，抽出される手順全体を見たときに手順が成立するための条件がある．このような条件を満たすためには，抽出される手順のグラフ上で整合性を取る必要があり，手順の抽出は本研究の評価として用いるコーパスとして適切である．

2.2 情報抽出

文献中に記述された情報を構造化して扱いやすくする情報抽出は，その目的に合わせて，様々な形式をとる．Kononova ら [11] は表形式に似た形式で材料の合成方法をまとめたコーパスを作成

表3: 合成手順コーパス [1] の辺ラベルの説明

辺ラベル	説明
Next_Operation	操作の進行
Recipe_Precursor	材料の投入
Recipe_Target	プロセスの最終生成物
Participant_Material	プロセス中の中間生成物
Solvent_Material	対象操作における溶媒
Atmospheric_Material	対象操作における雰囲気
Property_Of	材料に対する条件付け
Condition_Of	操作に対する条件付け
Number_Of	数値と単位の間の関係
Amount_Of	材料に対する量の条件付け
Descriptor_Of	条件付け
Type_Of	数値条件に対する条件付け
Brand_Of	材料や装置のブランド
Apparatus_Of	操作で利用する装置
Apparatus_Attr_Of	装置に対する数値条件付け
Coref_Of	同一の材料を表す

し，深層学習モデルと正規表現によるルールを組み合わせた手法で抽出に取り組んだ．Kuniyoshi ら [18] は全個体電池に関する文献に対してフローグラフのようにタグ付けしたコーパスを作成し，

深層学習モデルの固有表現抽出器とルールによる関係抽出を行った．イベント抽出は，文献中に記述された出来事を抽出するタスクである．Trieu ら [25] は，入れ子となっている出来事を抽出するための手法を提案した．Liu ら [26] は，機械の言語理解能力を確かめる Machine Reading Comprehension のタスクとして定式化して，生成したクエリに対する回答を文章中から探すことで抽出を進める手法を提案した．

このように，様々な情報抽出タスクが存在し，それぞれに対して，目的に特化した手法が提案されている．しかしながら，このように様々なタスクごとに特化した手法を開発するのは，人的や資金的なコストが高くなってしまうため非効率であり，これらを網羅的に抽出可能な手法があればコスト削減につながる．

2.3 関係分類

情報抽出タスクの一つに，関係抽出タスクがある [9,10,31]．関係抽出タスクは，2.1 節のプロセス抽出タスクと類似のタスクで，入力された文から用語を抽出し，その用語間の関係を抽出するタスクである．関係抽出では例えば，「Taro was hired by TTI.」という文を入力として受け取り，用語として用語ラベル「PERSON」の用語「Taro」と「ORGANIZATION」の「TTI」を抽出し，そのペア間の関係として関係ラベル「*BelongsTo*」の関係を抽出する，といったタスクである．このタスクでは，文献中に記述された情報を構造化することを目的として，様々な用語同士の関係性を抽出する．

既存の文間の関係抽出タスクはそのほとんどが遠距離教師学習 [21,32] のタスクで，教師あり学習のタスクの文間の関係抽出は少ない [20]．文間の関係抽出タスク用のコーパスを作成するコストは高く，共参照も考慮したタグ付けが必要となり，教師あり学習のコーパスとしての作成は困難であるためである．そのため，既存の文間の関係抽出手法はそのほとんどが，機械的に作成した教師データを用いて学習する遠距離教師学習を対象とした手法であり [19,33]，教師あり文間の関

係抽出手法はほとんど存在しない．Verga らは [19]Transformer を利用したモデルで，文献単位での文間の用語ペアに対する関係分類に取り組んだ．まず，用語の候補を探し，Transformer と畳み込みニューラルネットワークを用いて用語の表現を抽出して，得られた表現から全用語ペア間の関係を分類することで関係抽出を行う．Christopoulou らは [33] 用語を節点・関係を辺とした文書単位のグラフを作成し，その辺を分類することで，文間の関係を含む文書単位の関係抽出に取り組んだ．

プロセス抽出ではプロセスはほとんどの場合で複数文に渡って記述されるため，文間の用語同士の間の辺を抽出する必要がある．しかしながら，教師ありの文間の関係抽出手法が存在しないことから，関係抽出手法を単純に流用して行うことはできない．

関係抽出とプロセス抽出は文章を入力して用語を抽出し，用語間の関係を関連付ける点は共通している．異なる点は，関係抽出の目的が用語ペアの間の関係を抽出することであるのに対して，プロセス抽出は用語集合内で辺を張るグラフを抽出し，一続きのグラフを抽出することである．関係抽出では多くの場合で，ある用語ペアの間の関係はそのペアのみに着目して，他の用語との関係に影響を受けることは少なく，他の用語との関係を考慮した手法は少ない [34]．プロセス抽出では文献から抽出される情報全体で正しくなるように抽出する必要があり，関係抽出手法を単純に利用するだけでなく，プロセス抽出のための手法が必要である．

2.4 ニューラルネットワーク

ニューラルネットワークは人間の脳の神経細胞を模した数理モデルの総称である．深層学習はニューラルネットワークのモデルの一つで, 多層パーセプトロン（Multi-Layer Perceptron; MLP）や畳み込みニューラルネットワーク（Convolutional Nural Network; CNN）[35]，Transformer [24] などのニューラルネットワーク構造を多数積み重ねたものである．深層学習は内部の重み数が多いため，表現力が高い．本研究では深層学習を利用してモデルを作成する．

2.4.1 パーセプトロン

ニューラルネットワークの基本的なモデル化に，単純パーセプトロン [36] というモデルがある．入力のベクトルを $\boldsymbol{x} \in \mathbb{R}_{in}^{D}$，重み行列を $\boldsymbol{W} \in \mathbb{R}^{D_{in} \times D_{out}}$ として，バイアス項となるベクトルを $\boldsymbol{b} \in \mathbb{R}_{out}^{D}$ とすると，単純パーセプトロンの出力 $\hat{\boldsymbol{y}} \in \mathbb{R}_{out}^{D}$ は以下のように表現できる．

$$\hat{\boldsymbol{y}} = \boldsymbol{x}\boldsymbol{W} + \boldsymbol{b} \tag{2.1}$$

このうち，重み行列 $\boldsymbol{W}$ とバイアス $\boldsymbol{b}$ を学習することで，所望の出力が得られるようにモデルを最適化する．

単純パーセプトロンの問題点として，モデルの表現力が低く，線形分離可能な問題にしか適用できないという点がある．これを改善するために提案されたのが MLP である．MLP は非線形関数を用いた変換と，単純パーセプトロンでも行われている線形変換を繰り返すことで複雑な関数でも表現可能にするモデルである．入力のベクトルを $\boldsymbol{x} \in \mathbb{R}_{in}^{D}$，$l$ 層目の重み行列を $\boldsymbol{W}_l \in \mathbb{R}^{D_{hidden}^{(l-1)} \times D_{hidden}^{l}}$，バイアス項となるベクトルを $\boldsymbol{b} \in \mathbb{R}^{D_{hidden}^{(l)}}$，非線形関数を $f^{(l)}(\boldsymbol{x})$ とすると，L 層多層パーセプトロンの出力 $\hat{\boldsymbol{y}} \in \mathbb{R}_{out}^{D}$ は以下のように表現できる．

$$\hat{\boldsymbol{y}} = \mathrm{MLP}(x) = f^{(L)}(f^{(L-1)}(\cdots f^{(2)}(f^{(1)}(\boldsymbol{x}\boldsymbol{W}^{(1)} + \boldsymbol{b}^{(1)})\boldsymbol{W}^{(2)} + \boldsymbol{b}^{(2)})\cdots)\boldsymbol{W}^{(L)} + \boldsymbol{b}^{(L)}) \tag{2.2}$$

非線形関数は活性化関数と呼ばれ，繰り返し非線形関数が適用されることによって，非線形な出力が可能になる．活性化関数には様々な関数が利用されており，tanh 関数や，$\mathrm{ReLU}(x) = \max(0, x)$ [37] といった関数が代表的に用いられる．さらに，L 層目の出力層では出力のためにシグモイド関数や Softmax 関数，恒等関数といった関数を，モデル化する対象問題に合わせて利用する．例えば対象問題が回帰するタスクならば一般的に出力は実数の範囲での予測が必要となるため，恒等関数が利用されることが多く，多クラス分類であれば全クラスのうち個々のクラスの確率を出したいため，Softmax 関数が利用されることが多い．

2.4.2 誤差逆伝搬法

ニューラルネットワークは勾配法を利用して最適化する．ニューラルネットワークは層を積み重ねてモデルを作成し，教師データに沿うような出力が得られるようにパラメタを最適化するが，多層では解析的な解を得るのが困難になるため，勾配ベースの学習が必要となる．ニューラルネットワークは (2.1) 式でいう重み行列 $\boldsymbol{W}$ をパラメタとして最適化することで所望の出力を得られるようにする．ニューラルネットワークは一般的に，勾配法によって最適化される．勾配法は，対象の問題に合わせて設計される損失関数 $\mathcal{L}$ に対するパラメタの勾配を計算し，その勾配方向に更新をする操作を繰り返し行い最適化する．ある重み w に対して生じる更新は以下のように行われる．

$$w_{new} \leftarrow w_{old} + \mu \frac{\partial \mathcal{L}}{\partial w_{old}} \tag{2.3}$$

ニューラルネットワークの学習のための勾配は，層を積み重ねてモデルを作成する特性上，計算が煩雑になるため，単純に計算するのではなく，工夫されたアルゴリズムとして誤差逆伝搬法を利用して計算する．誤差逆伝搬法では，まず前向きにモデルの入力に対する出力を計算して各層における出力を記憶しておき，後ろ向きに層を遡るようにして，勾配の計算を進めて逆伝搬（バックプロパゲーション）させることで各パラメタの勾配を得る．

例として，2 層 MLP の場合の勾配を入出力を 1 次元とし，隠れ層の次元を $D_{hidden} = 1$ とした場合について計算してみる．2 層 MLP は以下のように表現できる．

$$z^{(1)} = f^{(1)}(x_1 \cdot w^{(1)} + b^{(1)}) \tag{2.4}$$

$$\hat{y} = f^{(2)}(z^{(1)} \cdot w^{(2)} + b^{(2)}) \tag{2.5}$$

得られた出力 $\hat{y}$ と教師データ y を比較して教師データとの差を示す損失関数 $\mathcal{L}(\hat{y}, y)$ として，パラメタに対する勾配を計算する．損失関数はタスクに応じて，尤度を最大化するように設計する．

誤差逆伝搬法によって，出力に近い層から順に勾配を計算する．まず出力に対する勾配 $\frac{\partial \mathcal{L}}{\partial \hat{y}}$ を計算する．この勾配については，損失関数が設計するものであることから既知であり，容易に計算可

能である．その後，順番に連鎖律を利用して勾配を計算する．まずは 2 層目の重み $w^{(2)}$ に対する勾配 $\frac{\partial \mathcal{L}}{\partial w^{(2)}}$ を計算し，1 層目の出力 $z^{(1)}$ に対する勾配 $\frac{\partial \mathcal{L}}{z^{(1)}}$，1 層目の重み $w^{(1)}$ に対する勾配 $\frac{\partial \mathcal{L}}{\partial w^{(1)}}$ と続けて計算を進める．

$$\frac{\partial \mathcal{L}}{\partial w^{(2)}} = \frac{\partial \mathcal{L}}{\partial y} \cdot \frac{\partial y}{\partial w^{(2)}} = \frac{\partial \mathcal{L}}{\partial y} \cdot \frac{\partial f^{(2)}(z^{(1)} \cdot w^{(2)} + b^{(2)})}{\partial w^{(2)}} \tag{2.6}$$

$$\frac{\partial \mathcal{L}}{z^{(1)}} = \frac{\partial \mathcal{L}}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial z^{(1)}} = \frac{\partial \mathcal{L}}{\partial \hat{y}} \cdot \frac{\partial f^{(2)}(z^{(1)} \cdot w^{(2)} + b^{(2)})}{\partial w^{(2)}} \tag{2.7}$$

$$\frac{\partial \mathcal{L}}{\partial w^{(1)}} = \frac{\partial \mathcal{L}}{\partial z^{(1)}} \cdot \frac{\partial z^{(1)}}{\partial w^{(1)}} = \frac{\partial \mathcal{L}}{\partial z^{(1)}} \cdot \frac{\partial f^{(1)}(x_1 \cdot w^{(1)} + b^{(1)})}{\partial w^{(1)}} \tag{2.8}$$

バイアス項の $b^{(1)}$ と $b^{(2)}$ も同様に計算される．

$$\frac{\partial \mathcal{L}}{\partial b^{(2)}} = \frac{\partial \mathcal{L}}{\partial y} \cdot \frac{\partial y}{\partial b^{(2)}} = \frac{\partial \mathcal{L}}{\partial y} \cdot \frac{\partial f^{(2)}(z^{(1)} \cdot w^{(2)} + b^{(2)})}{\partial b^{(2)}} \tag{2.9}$$

$$\frac{\partial \mathcal{L}}{\partial b^{(1)}} = \frac{\partial \mathcal{L}}{\partial z^{(1)}} \cdot \frac{\partial z^{(1)}}{\partial b^{(1)}} = \frac{\partial \mathcal{L}}{\partial z^{(1)}} \cdot \frac{\partial f^{(1)}(x_1 \cdot w^{(1)} + b^{(1)})}{\partial b^{(1)}} \tag{2.10}$$

このように，ニューラルネットワークでは誤差逆伝搬法を利用することで，効率的に勾配を計算することができる．

ニューラルネットワークの学習に用いるのは，勾配法の中でも，繰り返しデータの一部をサンプリングして学習を行う確率的勾配降下法（Stochastic Gradient Descent，SGD）を用いて学習する．確率的勾配降下法は学習時のデータの入力の仕方によって学習の進み方が異なる．入力の仕方は全部で三つあり，全てのデータを同時に入力して勾配を計算して全てのデータに対して同時に最適化を行うバッチ学習，一つのデータに対する最適化を繰り返し行うことで学習するオンライン学習，そしてこれらの間を取ってミニバッチとしていくつかのデータをサンプリングして学習を進めるミニバッチ学習がある．バッチ学習では全体に対して一気に最適化を行うため，局所解に陥ってしまう可能性が高く，オンライン学習では一つずつのデータに適応しようとするため，ノイズに脆弱である．それぞれの欠点を補うためにミニバッチ学習では，いくつかのデータのまとまりとして，ミニバッチを入力して学習させることで，これらの欠点のトレードオフを取ることができる．

2.4.3 Adaptive moment estimation (Adam)

勾配法では損失に対する勾配によって学習するが，(2.3) 式のように単純に勾配に沿って学習するだけでなく，収束を早めるための工夫が提案されている [38–40]．Sutskever ら [38] は (2.3) 式の SGD に対して，慣性に従った更新を行うための更新をすることで，局所解に陥ったりノイズの影響を小さくしたりするモーメンタム SGD を提案した．ある更新ステップ t の重みを w_t とし，勾配を $\frac{\partial \mathcal{L}}{\partial w_t} = g_t$ すると，モーメンタム SGD は以下の式で表現できる．

$$v_t = \beta v_t + (1-\beta) g_t \tag{2.11}$$

$$w_t = w_{t-1} + \mu v_t \tag{2.12}$$

これは，勾配の移動平均を取っており，モーメンタム SGD ではノイズのような急激な変化を抑えて学習を進められる．β の大きさに合わせてどれだけ慣性に従って学習を進めるか制御可能である．

Graves ら [39] は勾配の急激な変化を抑えるために，学習率を制御する RMSProp を提案した．RMSProp は以下のような式で表現できる．

$$v_t = \beta v_{t-1} + (1-\beta) g_t^2 \tag{2.13}$$

$$w_t = w_{t-1} - \frac{\alpha}{\sqrt{v_t + \epsilon}} g_t \tag{2.14}$$

RMSProp では，勾配が急激に変化したときに学習率が小さくなるようにして，振動やノイズといったような学習の妨げになるような更新を防ぐ．ϵ はゼロ除算を防ぐための小さい値である．

Kingma ら [40] はより収束しやすくするため，モーメンタム SGD と RMSProp を組み合わせ

た Adaptive moment estimation（Adam）を提案した．Adam は以下のような式で表せる．

$$m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t \tag{2.15}$$

$$v_t = \beta_2 v_{t-1} + (1-\beta_2) g_t^2 \tag{2.16}$$

$$\hat{m}_t = \frac{m_t}{1-\beta_1^t} \tag{2.17}$$

$$\hat{v}_t = \frac{v_t}{1-\beta_2^t} \tag{2.18}$$

$$\boldsymbol{w}_t = \boldsymbol{w}_{t-1} + \alpha \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon} \tag{2.19}$$

m_t はモーメンタム，v_t は RMSProp を表す項となっており，それらから更新量を定める．Adam は学習率の調整が自動で行われ，収束も早いことから利用性が高く，ニューラルネットワークの最適化手法ではデファクトスタンダードとなっている．β_1 と β_2 はハイパパラメタで，Adam で提案された論文ではそれぞれ $\beta_1 = 0.9, \beta_2 = 0.999$ が推奨されている．

2.4.4 ドロップアウト

ドロップアウト [41] は，ニューラルネットワークに用いられる正則化の手法の一つである．ドロップアウトでは学習時に計算される数値を学習のステップ毎にランダムで削除して利用可能な数値を限定することで，その数値に関わる重みの学習は行われなくなり，それぞれの数値が限られた数値のみで予測可能なように学習することで，モデルの汎化性能が向上する．

2.4.5 機械学習ライブラリ

近年の深層学習への注目に従い，ニューラルネットワークのような機械学習のためのライブラリが多く開発されている [42–44]．TensorFlow [43] は Google 社によって開発された深層学習用のライブラリで，計算グラフを構築して計算するように実装されている．

PyTorch [42] は，テンソルに対する計算を行う際に自動で計算グラフを構築し，ニューラルネットワークの学習時には誤差を逆伝搬させるための勾配を自動で微分して計算することができる．この特性から，Define by Run 型のフレームワークと呼ばれ，ネットワークを構築するに当たっ

て，順伝搬の計算を定義すると自動で逆伝搬の計算が可能になる．

2.5 Transformer

自然言語処理で用いられるニューラルネットワーク構造のうち，近年高い性能に寄与しているものとして，Transformer [24] がある．Transformer は注意機構を用いた構造である．

2.5.1 注意機構

注意機構は 2 つの系列の入力 $\boldsymbol{x} \in \mathbb{R}^{L^q \times D^q}$ とメモリ $\boldsymbol{m} \in \mathbb{R}^{L^m \times D^m}$ について，入力との関連度に基づいてメモリの情報を選択的に得るための手法である．ここで，L^i と L^m はそれぞれ入力とメモリの系列長で，D^i と D^m は表現ベクトルの次元数を表している．まず，重み $\boldsymbol{W}^q \in \mathbb{R}^{D^i \times D^h}$・$\boldsymbol{W}^k \in \mathbb{R}^{D^m \times D^h}$・$\boldsymbol{W}^v \in \mathbb{R}^{D^m \times D^h}$ を用いて，入力とメモリからクエリ $\boldsymbol{Q} \in \mathbb{R}^{L^i \times D^h}$・キー $\boldsymbol{K} \in \mathbb{R}^{L^m \times D^h}$・バリュー $\boldsymbol{V} \in \mathbb{R}^{L^m \times D^h}$ を計算する．

$$\boldsymbol{Q} = \boldsymbol{x}\boldsymbol{W}^q \tag{2.20}$$

$$\boldsymbol{K} = \boldsymbol{m}\boldsymbol{W}^k \tag{2.21}$$

$$\boldsymbol{V} = \boldsymbol{m}\boldsymbol{W}^v \tag{2.22}$$

クエリとキーを用いて，入力系列とメモリ系列の関連度として，各要素ごとの内積から注意の重み $\boldsymbol{A} \in \mathbb{R}^{L^i \times L^m}$ をとる．注意の重みは入力系列のある要素に対してメモリ系列のそれぞれの要素がどれだけ関連しているかどうかを示している．

$$\boldsymbol{A} = \mathrm{Softmax}\left(\frac{\boldsymbol{Q}\boldsymbol{K}^\mathsf{T}}{\sqrt{D^k}}\right) \tag{2.23}$$

この注意の重みを用いて入力系列に対して，関連度に基づいたメモリ系列の情報を付加したベクトル $\boldsymbol{O} \in \mathbb{R}^{D^i \times D^o}$ を重み行列 $\boldsymbol{W}^o \in \mathbb{R}^{D^h \times D^o}$ を用いて計算する．

$$\boldsymbol{O} = \boldsymbol{A}\boldsymbol{V}\boldsymbol{W}^o \tag{2.24}$$

このようにして，メモリ系列の情報を選択的に導入した入力系列の表現 $\boldsymbol{O}$ が得られる．注意機構をクエリ $\boldsymbol{Q}$・キー $\boldsymbol{K}$・バリュー $\boldsymbol{V}$ から以下のように定義する．

$$\boldsymbol{O} = \mathrm{Attention}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) \tag{2.25}$$

さらによりよい表現を得るために注意機構を拡張した，複数ヘッド注意機構がある．複数ヘッド注意機構では，入力系列とメモリ系列に対して，複数のクエリ $\boldsymbol{Q}$・キー $\boldsymbol{K}$・バリュー $\boldsymbol{V}$ を作成して得られた注意機構の出力をすべて結合したものである．複数ヘッド注意機構は以下のように表現できる．

$$\mathrm{MultiHeadAttention}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = \mathrm{Concat}(head^1, head^2, \cdots, head^H) \tag{2.26}$$

$$head^i = \mathrm{Attention}(\boldsymbol{Q}\boldsymbol{W}_i^q, \boldsymbol{K}\boldsymbol{W}_i^k, \boldsymbol{V}\boldsymbol{W}_i^v) \tag{2.27}$$

複数ヘッド注意機構は各ヘッドの次元数を減らして，複数の視点での系列同士の類似度に基づいてメモリ系列の表現を導入した入力を計算する．このことから，単純な注意機構と比較してより豊かな表現を得られる．

注意機構の利用方法として，自己注意機構がある．自己注意機構では入力とメモリを同一の系列を用いることで，その系列の表現を獲得するための手法である．つまり，自己注意機構は下式で表せる．

$$\mathrm{SelfAttention}(\boldsymbol{x}) = \mathrm{MultiHeadAttention}(\boldsymbol{x}, \boldsymbol{x}, \boldsymbol{x}) \tag{2.28}$$

2.5.2 モデルの構造

Transformer [24] は，自己注意機構を用いて表現を得る手法である．Transformer は自己注意機構を繰り返し用いて，入力系列の要素同士の複雑な関係を計算することで表現を計算する．さらに，複数層重ねた自己注意機構によって勾配が消失したりしないようにするために残差接続 [45] を用いる．自己注意機構は線形関数のみで定義されており，単体では単純な問題にしか用いることができないため，MLP を用いて複雑な計算を可能にする．これらは入力の系列の埋め込み $\boldsymbol{x}$ を

用いて，繰り返し回数を N として，$i = 1, 2, \cdots, N$ とすると，入力を符号化するエンコーダは下式で表せる．

$$\boldsymbol{z}_0 = \boldsymbol{y}_0 = \boldsymbol{x} \tag{2.29}$$

$$\boldsymbol{z}_i = \mathrm{Norm}(\boldsymbol{z}_{i-1} + \mathrm{SelfAttention}(\boldsymbol{z}_{i-1})) \tag{2.30}$$

$$\boldsymbol{y}_i = \mathrm{Norm}(\boldsymbol{y}_{i-1} + \mathrm{MLP}_i(\boldsymbol{z}_{i-1})) \tag{2.31}$$

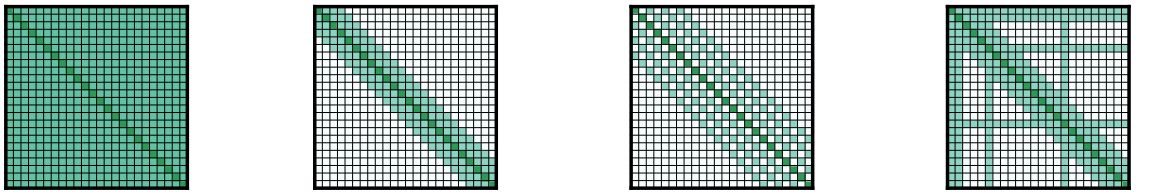
$$\tag{2.32}$$

Norm は正規化を表している．このように，Transformer は自己注意機構と MLP を組み合わせて繰り返し行うモデルである．繰り返しの自己注意機構によって，系列内の複雑な関係性に基づいた表現を得ることができる．

2.5.3 事前学習モデル

近年の自然言語処理分野では，大規模な文献から深層学習モデルを自己教師あり学習によって得られたモデルを，一般的な言語の情報を持ったモデルとして利用することで高い性能を示すようになっている [3, 22, 46–48]．事前学習モデルでは，文やトークンをベクトル化するために利用される．Peters ら [46] は大規模な文献で長・短期記憶（LSTM）[49, 50] を用いたモデルを，双方向に言語モデルを学習することで学習する Embeddings from Language Models（ELMo）を提案した．言語モデルは与えられた単語から次の単語を予測して，もともとの文を再現するタスクである．例えば「This is a pen.」という文が文献内にあった場合，順方向ではモデルはまず「This」を受け取って次の単語として「is」を出力し，「This」と自身の予測の「is」を受け取って「a」を予測して，続けて「pen」，「.」と予測し，逆方向では「.」から「pen」，「a」と先頭へ予測する．ELMo ではこの予測を双方向に行い，文の先頭と終わりから両者を予測するタスクを解くことで表現を獲得する．

Transformer モデルを利用した事前学習モデルは高い性能が得られることが確認されている．Transformer はコンピュータビジョン分野 [51] や自然言語処理分野のタスクに対する有効性が示



(a) Full n^2 attention (b) Sliding window attention (c) Dilated sliding window (d) Global+sliding window

図3: Longformer の Transformer を適用する範囲（引用：[3]）

されており，特に言語的な表現を得るための Transformer を利用した事前学習モデルでは豊富な表現を獲得可能である [3, 22, 47, 48].

Devlin ら [22] は ELMo より高い性能を持つモデルとして，Transformer ベースのモデルを用いて，大規模文献に対するマスク言語モデルの学習をする BERT（Bidirectional Encoder Representations from Transformers）を提案した．マスク言語モデルでは文の内の一部のトークンを MASK トークンに置き換え，置き換えられた MASK トークンから元のトークンを復元する，自己教師あり学習のタスクである．BERT は QA（Question Answering）タスクだけでなく用語抽出タスクなどでも既存の手法を大きく上回る高い性能を示した．

2.5.4 Longformer

Beltagy ら [3] は長い文長を扱うための Transformer ベースの事前学習モデルとして，Longformer を提案した．Transformer は文長に対して 2 乗のオーダーで計算量が増加する．このことから，BERT では文長が制限されてしまい，数文の入力が限界で，文献単位の入力はできない．例えば文をまたぐ文間の用語同士の関係抽出では，文献単位での抽出を行う必要があり，BERT で解くのは困難である．Longformer では文長に対して線形オーダーの計算量で処理可能なように工夫されており，長い文長を扱った計算が可能である．

線形オーダの計算量に制限するために，Longformer では Transformer を畳み込みニューラルネットワークを模して，図 3(b) のようにスライディングウィンドウを用いて，部分ごとに分けて Transformer を適用する．図 3は行と列それぞれがトークンを表していて，着色されている場合はそのセルの行と列のトークン同士で注意機構をとることを示している．

Longformer は，基本的に RoBERTa [47] を基準としたモデルで，RoBERTa の層をスライディングウィンドウを用いた Transformer に置き換えて作成する．スライディングウィンドウは入力に近い層から徐々にウィンドウサイズを大きくするようにして作成する．これは，入力に近い層で局所的な特徴を，出力に近い層では全体的な特徴を考慮するためで，ウィンドウサイズを固定したり，徐々に小さくするようにしたりして実験した場合では性能が低下する．

単純に系列をスライディングウィンドウによって部分に分けて Transformer を適用すると，局所的な特徴しか考慮できない．そこで，全体的な情報を考慮するために，図 3(d) のように一部の要素に対してはどのウィンドウに対しての計算のときにも結合して Transformer を適用する，全体注意機構を用いる．系列長を L とすると，図 3(a) のような単純に全体で注意機構を計算する Transformer の計算量は $\mathcal{O}(L^2)$ となり，ウィンドウサイズを w，スライドサイズを s とするとスライディングウィンドウを用いた場合の計算量は，$\mathcal{O}(w^2L/s)$ となる．Longformer では $s = w$ としているため，計算量は $\mathcal{O}(wL)$ となる．全体注意機構について考慮すると，全体注意機構の対象要素数を e とすると，$\mathcal{O}((w + e)L)$ となるが，全体注意機構をとる数 e は小さく，少数の要素のみに対してを対象とするため，$\mathcal{O}(wL)$ は保たれる．

Longformer では，Transformer モデル全体のうち一部のウィンドウに対して，図 3(c) のように，拡張畳み込み（Dilated Convolution）[52] を参考とした範囲で注意機構を利用する．拡張することで，同じ計算量でより遠いトークンとの関係性を考慮することが可能になる．ただし，全てのウィンドウを拡張すると性能が低下するため，一部のウィンドウのみを拡張する．

2.6 グラフ畳み込みネットワーク

Kipf ら [23] はグラフ構造の表現獲得のため，グラフ畳み込みネットワーク（Graph Convolutional Nerwork，GCN）を提案した．GCN はグラフ構造を，節点の表現を更新してグラフとしての表現を導入することで，ニューラルネットワークで扱えるようにする．

GCN は各節点の特徴を表すベクトルを辺に沿って伝播させることで各節点に周辺の構造情報を考慮した節点のベクトルが獲得できるため，グラフ構造を考慮した節点の表現を得ることがで

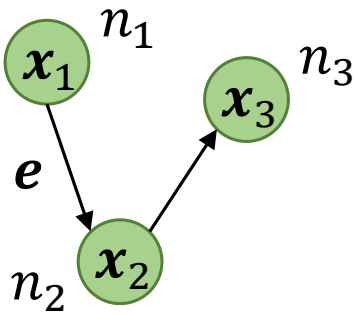


図4: グラフ構造

きる．図 4のようなグラフを考えるとき，それぞれの節点に ID を振り，ID が i の節点を n_i とし，それぞれの表現を $\boldsymbol{x}_i$，辺の表現を $\boldsymbol{e}$ とする．辺は (n_1, n_2) と (n_2, n_3) の二つであることから，一層の GCN では以下のように表現が更新される．ただし，$\cdot^{(t)}$ の t は GCN の層の数を表しており，$\otimes$ は要素積を示している．

$$\boldsymbol{x}_1^{(1)} = \boldsymbol{0} \tag{2.33}$$
$$\boldsymbol{x}_2^{(1)} = \boldsymbol{x}_1 \otimes \boldsymbol{e} \tag{2.34}$$
$$\boldsymbol{x}_3^{(1)} = \boldsymbol{x}_2 \otimes \boldsymbol{e} \tag{2.35}$$
$$\tag{2.36}$$

この式では，自身の表現が考慮できていない．そこで，自己ループを加えて，自身の表現を考慮できるようにすると，以下のように書き直せる．

$$\boldsymbol{x}_1^{(1)} = \boldsymbol{x}_1 + \boldsymbol{0} \tag{2.37}$$
$$\boldsymbol{x}_2^{(1)} = \boldsymbol{x}_2 + \boldsymbol{x}_1 \otimes \boldsymbol{e} \tag{2.38}$$
$$\boldsymbol{x}_3^{(1)} = \boldsymbol{x}_3 + \boldsymbol{x}_2 \otimes \boldsymbol{e} \tag{2.39}$$
$$\tag{2.40}$$

$\boldsymbol{x}_2^{(1)}$ を見ると，n_1 からの接続があることから，n_1 の表現 x_1 が考慮できているとわかる．このように，GCN を利用して表現を更新することで，自身の表現に対してグラフ構造の表現を導入することができる．

GCN では層を増やすとより離れた節点の表現を考慮することができる．二層の GCN の場合で

は，表現は以下のようになる．

$$\boldsymbol{x}_1^{(2)} = \boldsymbol{x}_1^{(1)} + \boldsymbol{0} = \boldsymbol{x}_1 \tag{2.41}$$

$$\boldsymbol{x}_2^{(2)} = \boldsymbol{x}_2^{(1)} + \boldsymbol{x}_1^{(1)} \otimes \boldsymbol{e} = 2\boldsymbol{x}_1 \otimes \boldsymbol{e} + \boldsymbol{x}_2 \tag{2.42}$$

$$\boldsymbol{x}_3^{(2)} = \boldsymbol{x}_3^{(1)} + \boldsymbol{x}_2^{(1)} \otimes \boldsymbol{e} = \boldsymbol{x}_1 \otimes \boldsymbol{e} + 2\boldsymbol{x}_2 \otimes \boldsymbol{e} + \boldsymbol{x}_3 \tag{2.43}$$

$$\tag{2.44}$$

二層の GCN を適用した n_3 の表現 $\boldsymbol{x}_3^{(2)}$ を見ると，辺が二つ分離れている n_1 の表現 $\boldsymbol{x}_1$ が導入されていることがわかる．このように層の数によって考慮可能な節点の範囲が定められる．

3. 提案手法

本研究では，情報抽出をグラフの抽出として定式化し，図 5のように既存のシステムで抽出したグラフを事後編集して文脈上とグラフ上の両者の観点での整合性を取ったグラフを抽出する．グラフの編集は，グラフの辺分類器によって節点ペア間の辺のクラスを分類し，その結果で上書きすることで行う．グラフ構造は多くの情報を包括的に表現可能なデータ構造であり，グラフの抽出が可能であれば，様々な形式の情報抽出に利用可能になる．本研究ではグラフの編集の中でも，編集が容易な辺のみを対象として抽出する．モデルは文脈情報を導入するために事前学習モデル Longformer [3] を利用して節点の表現を作成して，グラフ上で周辺の節点の情報を導入するために，節点の情報を接続している辺に沿って伝搬させる手法である GCN を利用する．

3.1 タスクの設定

本研究で対象とするのは文献中からの情報抽出で，情報をグラフ構造に変換することで，グラフの抽出として定式化する．グラフ構造は多くの情報を網羅的に扱うことが可能な形式で，例えばフローチャートのような順次構造や表形式の情報などの，多岐にわたる情報を表現可能であり，多くの情報抽出タスクにおける抽出対象の情報を表現可能である．このことから，グラフ構造の情報の抽出として定式化して，その問題を解くことができる手法があれば，所望の情報をグラフ構造に変換してその手法を適用することで抽出可能になる．

本研究ではグラフ構造の抽出のうち，辺の抽出を対象に取り組む．本研究ではグラフを編集して抽出を行う．しかしながら，辺の編集は自由に編集することが可能であるが，節点を編集する場合，辺は節点に依存して定められるため，その編集された節点とその節点以外のグラフとの間で整合性を保った編集をするのは難しい．例えば，ある節点 A と節点 B の間に辺が存在するときに，節点 A を別の節点 C に置換する場合では節点 A と節点 B の間に存在した辺は節点 C に対する辺ではないため，辺を正しく処理しなければ不整合が生じる．このような問題から，節点の編集は困

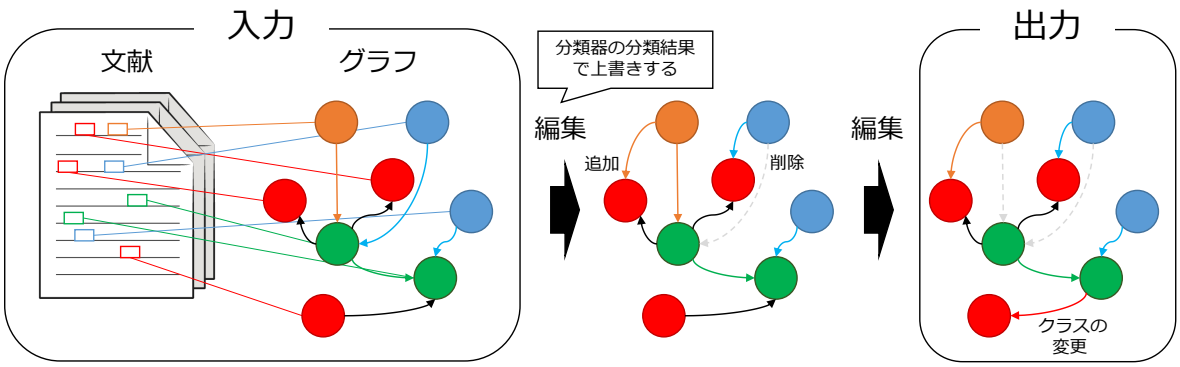


図5: 提案手法の全体像

難であることから，まずは辺の編集に取り組む．

文献からのグラフの抽出タスクを定義する．本研究の対象は文献からのグラフ形式の情報抽出で，本研究で対象としている辺の抽出に入力として利用可能なものは，抽出対象の文献の文章情報 text と，文章に直接タグ付けされている節点となる用語 $\mathcal{N} = [n_1, n_2, \cdots, n_{|\mathcal{N}|}]$ の二つである．用語は文章中のどの語句かを示す位置とその用語の種類を示す用語ラベルによって定義されている．文章情報と用語の情報から，その用語間の関係性を示す辺集合 $\mathcal{E}$ を抽出して出力することが本タスクの目的である．

3.2 グラフの編集モデル

グラフの編集モデルでは，抽出対象の文章から既存のシステムなどから得られたグラフ $\mathcal{G}(\mathcal{N}, \mathcal{E})$ を入力として，文章の面と抽出されるグラフ全体の両者を考慮した上での整合性が取れるように入力されたグラフを編集して，辺を編集した後のグラフを出力する．$\mathcal{N}$ は既存のシステムによって抽出された節点集合，$\mathcal{E}$ は辺の集合である．編集する対象は辺のみであることから，出力されるグラフの節点は入力されたグラフの節点と同一である．

編集はそれぞれの辺のクラスを，3.2.1 節の深層学習を用いたグラフの辺分類モデルによって順番に上書きするように進める．グラフの辺分類モデルでは，文脈と抽出されるグラフ全体の整合性を考慮した表現を利用して，任意の節点間の辺のクラスを分類する．このモデルを利用して，全組み合わせの節点ペア間の辺を分類することで編集を進める．編集する順序は，簡単な問題から

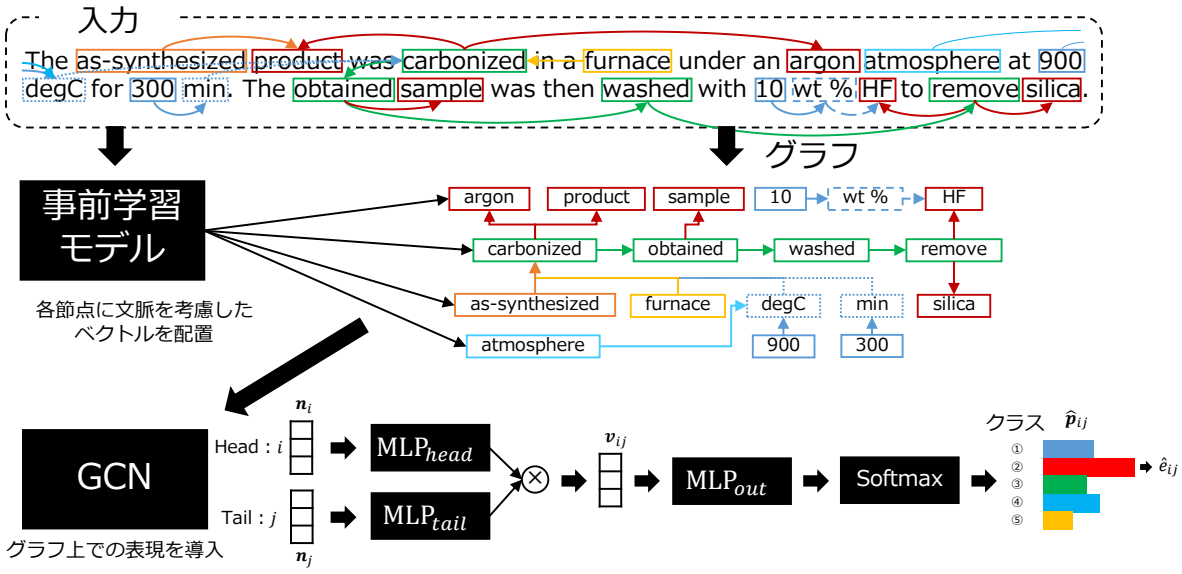


図6: グラフの辺分類モデルの模式図

難しい問題へ取り組むように解くことで難しい問題が解きやすくなる，という人間の知見 [53, 54] から，文章中での節点を出現順序に並べたときに隣り合っている節点ペアから順番に編集し，徐々に離れている節点ペアの間の辺を編集する．

グラフの辺分類モデルは，文脈を考慮するために Transformer [24] ベースの事前学習モデルを用い，プロセスの整合性を考慮するためにグラフの構造を GCN を用いたネットワークを利用して，グラフ全体の整合性を考慮した表現を盛り込む．

3.2.1 グラフの辺分類モデル

グラフの辺分類モデルとして，任意の節点ペア (n_i, n_j) 間の辺を分類するモデルを提案する．モデル EdgeClassifier は図 6のように，入力として文章の情報 text とグラフ $\mathcal{G}(\mathcal{N}, \mathcal{E})$，編集対象の節点ペアの ID$(i, j)$ を受け取り，編集対象の節点ペア間の辺のクラス $\hat{e}_{ij}$ を出力する．ここでは，便宜上，文章から各節点の表現を作成する Encode を分割して，文章 text と節点の情報 $\mathcal{N}$ から各節点の文約を考慮したベクトルを得る Encode(text, $\mathcal{N}$) と，その表現を導入したグラフ $\mathcal{G}(\bar{\mathcal{N}}, \mathcal{E})$ と編集対象の節点ペアの ID(i, j) を入力して編集した辺を出力する EdgeClassifier$(\mathcal{G}(\bar{\mathcal{N}}, \mathcal{E}), (i, j))$

によって表現できる．これは以下の式で表せる．

$$\bar{\mathcal{N}} = \mathrm{Encode}(\mathrm{text}, \mathcal{N}) \tag{3.1}$$

$$\hat{e}_{ij} = \mathrm{EdgeClassifier}(\mathcal{G}(\bar{\mathcal{N}}, \mathcal{E}), (i, j)) \tag{3.2}$$

モデルでは文脈の整合性を保つために Transformer ベースの事前学習モデルによる特徴を利用し，プロセスの整合性を保つために GCN を利用する．まず節点となる用語のベクトルを文脈を考慮したベクトルに埋め込み，文脈情報を表現する．さらにグラフ上での整合性を考慮するため，入力されたグラフ上の節点に対して対応する用語の表現ベクトルを配置し，GCN を利用して，グラフ上での前後関係などの情報を盛り込んだベクトル表現を得る．得られたベクトル表現は，文脈情報とグラフの情報を同時に考慮した表現であり，この表現から用語ペア間の辺について分類することで整合性をとる．GCN に入力するグラフの辺は，初期値は既存のシステムで抽出した辺，その後は編集過程の辺を入力として，その編集時点でのグラフの構造を考慮した辺の編集が進行していくことに注意する．

まず入力された文章 text を Longformer [3] のような Transformer ベースの事前学習モデルを用いて，文脈を考慮したベクトル表現へと埋め込む．Transformer ベースの事前学習モデルでは，単語を更に分割したサブワード単位に表現を計算する．節点になる用語の表現へ落とし込むため，得られたサブワード単位の埋め込みに対して，サブワードの表現の各次元で最大値をとるプーリング（Pool）をして用語レベルの特徴 $\boldsymbol{v}^{text} \in \mathbb{R}^{L \times D^{text}}$ にする．D^{text} は埋め込みの次元数，L は用語の数である．

$$\boldsymbol{v}^{text} = \mathrm{Pool}(\mathrm{Transformer}(\mathrm{text})) \tag{3.3}$$

各用語の特徴に対して，次元を D^{label} とした用語ラベルの埋め込み $\boldsymbol{v}^{label} \in \mathbb{R}^{L \times D^{label}}$ を結合し，それぞれの節点の埋め込み $\boldsymbol{n} \in \mathbb{R}^{L \times D}$ とする．ある節点 n_i の埋め込み $\bar{\boldsymbol{n_i}}$ は以下の様に表現できる．

$$\bar{\boldsymbol{n_i}} = [\boldsymbol{v}_i^{text}; \boldsymbol{v}_i^{label}] \tag{3.4}$$

この埋め込みを含めた節点集合を $\bar{\mathcal{N}}$ とする．ここまでの Transformer による表現抽出とプーリングによる節点の文脈表現と，ラベルの表現を合わせた操作で $\bar{\mathcal{N}}$ が得られるまでが $\mathrm{Encode}(\mathrm{text}, \mathcal{N})$ の関数である．

用語は節点とするため，各用語のベクトルをグラフの節点に配置し，GCN によって入力された辺に沿って節点の表現を伝搬させることで，グラフ上での整合性を考慮した節点の表現を計算する．M 層の GCN によってグラフから抽出した節点の表現ベクトル集合 $\mathcal{N}_M \in \mathbb{R}^{L \times D}$ は，$i = 0, 1, \cdots, M$ としたときに以下の漸化式によって得られる．

$$\mathcal{N}_1 = \mathrm{GCN}(\bar{\mathcal{N}}, \mathcal{E}) \tag{3.5}$$

$$\mathcal{N}_{m+1} = \mathrm{GCN}(\mathcal{N}_m, \mathcal{E}) + \mathcal{N}_m \tag{3.6}$$

GCN は複数層重ねることで，辺のつながりが隣接の節点だけでなく，M 個離れた節点の情報まで着目することができ，グラフ上で離れた部分の情報も考慮することが可能になる．事前学習モデルによって文脈を考慮した表現を獲得し，GCN によって得られた表現 $\mathcal{N}_M$ はグラフ上で隣接 M 個の節点の情報が考慮できており，グラフ上での表現と文脈上での表現の両者を含む表現である．一つ前の GCN の結果を足し上げているのは，勾配消失の防止のためで，残差接続 [45] と呼ばれる．

GCN から得られたグラフの情報と文脈情報を含む節点の表現 $\mathcal{N}_M$ より，辺のクラス分類をする．入力された編集対象の節点 n_i, n_j 間の辺 e_{ij} の分類をする場合を考えると，GCN から得られた表現 $\mathcal{N}_M$ から n_i, n_j 間の辺の表現として，$\boldsymbol{v}_{ij}$ を計算する．追加の表現として，$\boldsymbol{v}_{ij}$ には対象節点ペアが文献上で隣接しているかどうか，Head と Tail の出現順序が Head が先か Tail が先かを示す表現 $\boldsymbol{b}_{ij}$ を結合する．各節点が Head になる場合と Tail になる場合の空間に写像するための全結合層を MLP を用いてそれぞれ $\mathrm{MLP}_{head}, \mathrm{MLP}_{tail}$ とすると，$\boldsymbol{v}_{ij}$ は以下のように表現できる．

$$\boldsymbol{v}_{ij} = [\mathrm{MLP}_{head}(\boldsymbol{n}_i) \otimes \mathrm{MLP}_{tail}(\boldsymbol{n}_j); \boldsymbol{b}_{ij}] \tag{3.7}$$

$\otimes$ は要素積を示しており，同一次元の $\boldsymbol{x}$ と $\boldsymbol{y}$ の要素積 $\boldsymbol{x} \otimes \boldsymbol{y}$ の l 番目の要素は $x_l \times y_l$ のように $\boldsymbol{x}$ と $\boldsymbol{y}$ の l 番目の要素の積となっている．

さらに，この表現から予測を得るために結合層 $\mathrm{MLP}_{out}(\boldsymbol{x})$ を用いて，辺無しのクラスを含めたクラス分類をし，節点 n_i, n_j 間に辺クラスが割り当てられる確率 $\boldsymbol{p}_{ij} \in \mathbb{R}^{|\mathcal{C}|}$ を計算する．$\mathcal{C}$ は辺無しのクラスを含む辺のクラスの集合である．

$$\boldsymbol{u} = \mathrm{MLP}_{out}(\boldsymbol{v}_{ij}) \tag{3.8}$$

$$\hat{\boldsymbol{p}}_{ij} = \mathrm{Softmax}(\boldsymbol{u}) = \frac{\exp(\boldsymbol{u})}{\sum_{l=1}^{R} \exp(u_l)} \tag{3.9}$$

得られた $\boldsymbol{p}_{ij}$ に基づき，最大の確率を示すクラスを予測とすることで，所望の節点 n_i, n_j 間の辺の予測ラベル $\hat{e}_{ij}$ が得られる．

$$\hat{e}_{ij} = \arg\max \hat{\boldsymbol{p}}_{ij} \tag{3.10}$$

テキスト情報を導入した $\bar{\mathcal{N}}$ から $\hat{e}_{ij}$ を得るまでが $\mathrm{EdgeClassifier}(\mathcal{G}(\bar{\mathcal{N}}, \mathcal{E}), (i, j))$ の関数で行われる操作である．

3.2.2 グラフの編集

グラフの編集は，入力されたグラフの節点ペアの全組み合わせに対して，3.2.1 節の辺分類モデルによる分類結果によって上書きして行くことで進める．編集の順序は，簡単な問題を先に解き，難しい問題は後に解くと推論しやすい，というヒューリスティクスに基づき，節点の元の用語同士が文章中で近いものから編集を進め，徐々に離れた節点同士の編集を行う．用語の距離は出現順序に基づいて計算し，用語を出現順に並べてその順番をもとに，例えば文章中で m 番目に出現した用語と $m+3$ 番目に出現した用語同士の距離は 3 とする．

順番に編集するとき，距離が同一のものについては同時に編集する．グラフの辺の編集モデルは，大きく分けて三つの部分に分けることができる．一つ目が事前学習モデルによる文脈を考慮した節点のベクトル表現の作成，二つ目が GCN を用いた節点のベクトルに対するグラフの表現の導入，三つ目が得られた文脈とグラフの表現を含むベクトルから節点ペア間の辺の予測である．この内，全ての編集の段階において文章が変化しないため，一つ目は編集の全体を通して一度のみ計算すればよい．しかしながら，二つ目と三つ目は，グラフが編集が行われる度に変化すること

Algorithm 1: グラフの編集手順

Distance($\bar{\mathcal{N}}, d_1, d_2$)：距離が d_1 以上 d_2 未満の節点ペア集合 $\mathcal{P}$ を返す関数

Encode(text, $\mathcal{N}$)：文章 text 上の節点の位置とラベル情報を含むの節点集合 $\mathcal{N}$ から節点の埋め込み集合 $\bar{\mathcal{N}}$ を返す関数

EdgeClassifier($\bar{\mathcal{N}}, \mathcal{E}, p$)：各節点に表現ベクトルが配置されたグラフ $\mathcal{G}(\bar{\mathcal{N}}, \mathcal{E})$ から節点ペア $p = (i, j)$ の間の辺 e_{ij} を予測する分類器

Edit($\mathcal{E}, p, c$): 節点ペア p の間の辺のクラスをクラス c に編集した辺集合を返す関数

Input: $text$：文章

$\mathcal{N}$：節点の集合で $text$ 中の用語の位置とラベルの情報

$\mathcal{E}$：他のシステムで抽出された，もしくは空の辺集合

d_{max}：距離の最大値

Output: $\mathcal{E}$：辺集合

Initialize: $\mathcal{E}' \Leftarrow \mathcal{E}$

$\bar{\mathcal{N}} \Leftarrow$ Encode(text, $\mathcal{N}$)

while d in range(max($|\bar{\mathcal{N}}|, d_{max}$)) **do**

  if $d = d_{max}$ **then**

    $\mathcal{P} \Leftarrow$ Distance($\mathcal{N}, d_{max}, \infty$)

  else

    $\mathcal{P} \Leftarrow$ Distance($\mathcal{N}, d, d+1$)

  end if

  while p in $\mathcal{P}$ **do**

    $e_{ij} \Leftarrow$ EdgeClassifier($\mathcal{N}, \mathcal{E}, p$)

    $\mathcal{E}' \Leftarrow$ Edit($\mathcal{E}', p, c$)

  end while

  $\mathcal{E} \Leftarrow \mathcal{E}'$

end while

から，毎回再計算する必要があり，毎回再計算した場合では全節点ペアの組み合わせ分の $|\mathcal{N}|^2$ 回の計算をしなければならない．そこで計算の効率化のため，順番に編集するとき，距離が同一のものについては同時に編集する．同時に編集することで，二つ目の GCN は $|\mathcal{N}-1|$ 回の計算回数まで抑えることができる．ただし，三つ目の辺の予測はすべてのペアに対して予測しなければならないため，計算回数は $|\mathcal{N}|^2$ であるが，同一距離のペアについては並列計算が可能になり，計算は効率化できる．

さらに，効率的に計算するために，距離の上限値 d_{max} を設けておき，d_{max} 以上の距離の節点

ペア間の辺はすべて同時に編集を行う．同一距離のものを同時に編集しても $|\mathcal{N}|-1$ 回の編集が必要で，実際に適応する情報抽出の問題に合わせると，[（文章中に存在する用語数）−1] の編集が必要である．例えば合成手順コーパスに対するグラフの抽出では平均で 80 回以上，最大で 300 回以上の編集が必要となる．実際に計算機で学習する場合を考えると，GCN の計算は $N-1$ 回必要であり，計算するためには計算時間が大量に必要で，その計算の勾配を保持するための計算リソースも膨大に必要になってしまう．これを防ぐために，距離の上限値 d_{max} を設けておき，d_{max} 以上の節点ペア間の辺は同時に編集を進めることでこれを解決する．これは，遠くで述べられている用語同士に関連性がある場合が少ないことから，遠い部分については個別に計算する必要が無いためで，例えば離れた文に記述された節点ペアでは係り受けなどの関連性は少ない．

具体的な編集方法をアルゴリズム 1に示した．まず，グラフの辺の初期値をルールベースで抽出した辺にし，距離が近い用語ペアの順番に，同一距離ごとに GCN を用いたネットワークによってクラス分類をして編集を進める．

3.2.3 学習

モデルの学習は，辺分類器が正しく分類できるように，対数尤度を最大化する．この時，グラフの編集は自身の出力をそのモデル自身に入力する自己回帰モデルとなっているが，編集の際には予測された各クラスの確率から最大確率の辺をサンプリングしているため，勾配は各編集ステップで独立した勾配になっていることに注意しなければならない．また，各編集ステップで計算する必要があるのは辺が編集されたときに変化する計算のみであるため，勾配を計算するのは GCN による計算以降の部分のみであり，それ以前の部分については勾配を再計算する必要はない．

正解の辺を e_{ij} とすると，損失 $\mathcal{L}$ は以下のように定義できる．

$$\mathcal{L} = -\sum_{i=1}^{|\mathcal{N}|}\sum_{j=1}^{|\mathcal{N}|}\sum_{c\in\mathcal{C}} \mathbb{1}[c = e_{ij}] \cdot w_c \log p_{ij} \tag{3.11}$$

ただし，$\mathbb{1}[\cdot]$ は，括弧内の条件を満たしたときに 1，満たさないときには 0 を返す関数で，w_c はクラスごとの重みである．クラスごとの重み w_c は学習の仕方を制御するためのもので，例えば特定

のクラスに着目して学習を進めたい場合などに利用可能である．$w_c = 1$ とすると，この損失は交差エントロピーを表しており，問題が多クラス分類の場合に最尤推定をしたときに得られる目的関数である．この損失に基づいて，確率的勾配法によってモデルのパラメタを更新することで学習する．

4. 実験と考察

本章ではグラフの編集モデルを評価し，さらにモデルの効果について調査する．4.1 節では実験を行う環境やモデルの設定といった実験設定について述べる．次にベースラインとして用いるルールベースについて 4.2 節で述べる．4.3 節では得られた結果とそれに対する考察を述べる．

4.1 実験設定

実験のための環境や実験に利用したコーパスなどの詳細について述べる．

4.1.1 実験環境

実験を行うための実装はプログラミング言語 Python version 3.7.7 を用いて実装した．深層学習モデルを実装するための機械学習フレームワークは PyTorch version 1.5.0 [42] を用いた．さらに GCN の実装のため，グラフを用いたニューラルネットワークに関するライブラリ PyTorch Geometric [55] を用いて実装した．

実験には複数の計算機を利用しており，表 4に示したものを用いて実験した．表 4には中央演算処理装置（Central Processing Unit; CPU），図学処理装置（Graphics Processing Unit; GPU），主記憶装置（主メモリ）を示した．利用する計算機ごとに結果が変化することはないため，計算は実験時に計算リソースを利用可能なもののうち，利用しやすいものを選択して利用した．

表4: 計算環境

CPU	GPU	主メモリ	台数
Intel(R) Xeon(R) CPU E5-2698	TESLA V100 × 4	256GB	1
Intel(R) Core(TM) i9-10900K CPU	GeForce RTX 3090	64GB	4
Intel(R) Core(TM) i7-5960X CPU	GeForce GTX TITAN X (Pascal) × 3	64GB	2
Intel(R) Core(TM) i9-7900X CPU	TITAN V × 2	128GB	3

4.1.2 コーパス

本実験で利用するコーパスは，2.1 節で述べた合成手順コーパス [1] を利用し，無機材料文献からの合成プロセスの抽出に対する抽出性能で評価する．合成手順コーパスでタグ付けされている合成プロセスは，操作の進行や材料の投入，条件付けといったような辺が用いられている．提案手法は文脈とグラフの両者の整合性をとる編集を行う手法である．合成プロセスの抽出では，文脈上の整合性を考慮すると，係り受けといったような情報から，材料 A を操作 X する，といった文から A と X の間の辺を推論可能である．さらにグラフ上の表現を利用することで，例えば生成物に対する辺のクラスを分類するときには，それまでに行われた操作や投入された材料などの条件を考慮する，といったようなプロセスの全体として考えたときの表現を利用することができる．

合成手順コーパスの統計量は表 5に示し，節点と辺の出現数は，訓練データと開発データを合わせたものから算出した数値を，それぞれ節点は表 6，辺は表 7に示した．統計量は以下の数値を示した．

- データ数

  訓練用・開発用・評価用それぞれに対するデータの数を示す．それぞれのデータは文献と

表5: 合成手順コーパスの統計量

訓練用データ数	199
開発用データ数	15
評価用データ数	15
平均節点数	86
最大節点数	323
平均辺数	81
最大辺数	323
平均文章長	237
最大文章長	731
平均トークン数	350
最大トークン数	1,036

その文献に対してタグ付けされているグラフを表す．データの分割は，コーパスを作成した Mysore らが分割して公開したものと同一のものを用いて実験を行う．

- 節点数

  文献にタグ付けされたグラフにおける節点の数を示している．各節点は用語を示していることから，文献にタグ付けされた用語数と一致する．訓練用データと開発用データを合わせたものから平均と最大値を算出した．

- 辺数

  文献にタグ付けされたグラフにおける辺の数を示している．各辺は節点ペア間の関係を示していることから，文献にタグ付けされた関係数と一致する．訓練用データと開発用デー

Material	4,548
Operation	3,461
Number	3,096
Condition-Unit	1,464
Amount-Unit	1,289
Material-Descriptor	1,281
Property-Misc	506
Condition-Misc	500
Synthesis-Apparatus	453
Nonrecipe-Material	362
Brand	321
Apparatus-Descriptor	175
Amount-Misc	163
Meta	140
Property-Type	134
Condition-Type	121
Reference	116
Property-Unit	99
Apparatus-Unit	95
Characterization-Apparatus	56
Apparatus-Property-Type	26

表6: 合成手順コーパスにおける訓練データと開発データに出現する節点のクラスごとの合計数

タを合わせたものから平均と最大値を算出した．

- 文章長

  文献の単語数を示している．単語数は SciSpaCy [56] の en_core_sci_sm モデル[*3]を用いて単語分割して計算する．訓練用データと開発用データを合わせたものから平均と最大値を算出した．

- トークン数

  Longformer によって計算することため，Longformer で利用したサブワード分割における文長を示す．訓練用データと開発用データを合わせたものから平均と最大値を算出した．

合成手順コーパスの訓練データ内に一件，1,680 の節点を持つ突出した文献が存在し，次に多い文献で節点の数が 323 であったことから，この文献については特殊な文献として扱い，除外して 199 件の訓練データで学習を行った．訓練データはモデルの学習に利用し，開発データはモデルの学習

[*3] https://s3-us-west-2.amazonaws.com/ai2-s2-scispacy/releases/v0.3.0/en_core_sci_sm-0.3.0.tar.gz

Next_Operation	3,082
Number_Of	3,024
Condition_Of	1,942
Participant_Material	1,836
Amount_Of	1,642
Descriptor_Of	1,586
Recipe_Precursor	943
Property_Of	621
Solvent_Material	491
Apparatus_Of	475
Brand_Of	465
Recipe_Target	394
Coref_Of	279
Atmospheric_Material	194
Type_Of	171
Apparatus_Attr_Of	96

表7: 合成手順コーパスにおける訓練データと開発データに出現する辺のクラスごとの合計数

自体には利用せずに，学習に利用されていないデータでの挙動を確認する指標として利用し，評価データは学習や指標として全く利用せずに新規のデータとして評価のために利用した．BERT [22] や RoBERTa [47] といったような，一般的な Transformer ベースのモデルが受け取れる入力のトークン数である 512 で，最大のトークン数 1,036 はこれを超えているため，長い文献を扱うことが可能なモデルを利用する必要がある．

4.1.3 グラフの編集モデルの設定

実験において用いるグラフの編集モデルの設定について述べる．表 8に示したパラメタは全ての実験で統一して利用するパラメタで，事前実験によって高い性能が得られたパラメタを利用した．抽出対象が系列長が長い文献単位のグラフの抽出であるため，事前学習モデルには長い系列を扱うことができる Longformer [3] を用いた．最適化には Adam [40] を用いており，Adam のパラメタである学習率，β_1，β_2 はそれぞれ，著者が推奨するデフォルト値を利用した．利用した MLP の層の数はすべて 3 層にした．

表 9に示したハイパパラメタについては，パラメタの組み合わせを全探索したときの最適なパラメタを選択した．全探索はどのモデルも十分収束する 130 エポックで学習を行い，モデルの学習が終了したとき，開発データに対して最も高いマイクロ F 値を記録したエポックのときのモデル

表8: すべての実験で共通するグラフの編集モデルの設定

ハイパパラメタ名	パラメタ
事前学習モデル	Longformer (allenai/longformer-base-4096) [3]
GCN	Kipf らの手法 [23]
最適化手法	Adam
学習率	0.001
β_1	0.9
β_2	0.999
MLP_{head} の層の数	3
MLP_{tail} の層の数	3
MLP_{out} の層の数	3
$\boldsymbol{u}$ のドロップアウト率	0.3

をその試行のモデルとし，最大のスコアを記録したモデルを評価に用いた．探索して得られたパラメタは表 10に示した．このパラメタのチューニングについてのすべての結果は付録 B に示し，チューニングに基づく考察は 4.3.3 節に示した．

事前実験によって損失における w_c を $w_c = 1$ として学習した場合では，予測する辺の数のうちのほとんどが負例であることから，モデルが負例しか予測しなくなってしまったため，正例のクラスの損失が大きくなるようにした．学習初期には正例の損失が大きくなるように係数を掛けておき，学習が進むにつれ負例と同一の重み付けになるように係数をスケジューリングする．本研究で対象としているタスクは，文献単位でのグラフ抽出における辺の抽出である．抽出対象のグラフが有向グラフであることから，$2|\mathcal{N}|^2$ 個の辺を予測する必要があるが，辺を持たない節点ペアが多くを占めていて，正例となる辺は極めて小数である．そのため，事前実験では $w_c = 1$ で学習すると，負例に対する損失の信号が大きくなり，ほぼすべての辺を負例として予測してしまうように学習が進み，その状態で飽和してしまう．そこで学習初期には正例に対する損失の重みを大きくしておき，学習の進行に合わせて正例の損失の重みを，負例に対する重みと同じ数値になるようにスケジューリングする．クラスごとの重み w_c は，その時点でのエポック数を #EPOCH,

表9: 探索するパラメタ

ハイパパラメタ名	探索対象のリスト
GCN の層の数	0, 1, 2, 3
次元数	64, 128, 256
d_{max}	1, 5, 10, 15

表10: 探索によって得られた最適なパラメタ

ハイパパラメタ名	パラメタ
GCN の層の数	2
次元数	128
d_{max}	15

負例のクラスを c_{neg} として，下式の様にした．

$$w_c = \begin{cases} \max(0.01 \cdot \#\mathrm{EPOCH}^{1.5}, 1.0) & \text{if } c = c_{neg} \\ 1.0 & \text{otherwise} \end{cases} \tag{4.1}$$

この式では，負例の重みが 0.01 から徐々に指数関数に従って最大値 1.0 まで増加し，1.0 になって以降は正例の重みが 1.0 になる．この関数にした理由は，学習データにおける正例の数の割合が予測する辺の数の約 1% であったため，正例と負例の損失の強度をほぼ同程度になるように学習を始め，徐々に本来の損失の強度に戻して学習を行うようにするためである．

4.1.4 評価方法

評価は合成手順コーパスの開発データに対してプロセスを抽出した性能を評価することで行う．提案手法は辺の抽出に対する手法である．辺の抽出性能の評価を行うため，節点となる用語の位置とラベルは正解を与えた状態で，辺の抽出のみを対象として評価を行う．比較対象は四種類のモデルを利用し，ベースラインである 4.2 節のルールベースの抽出器（ルールベース），この抽出器によって抽出されるグラフを編集モデルで編集したもの（ルール＋編集），辺が無い状態のグラフから編集モデルによって辺を接続したもの（編集のみ），ランダムに辺を接続したもの（ランダム＋編集），編集順序をランダムにしたもの（ランダム順序＋編集）を比較し評価する．ランダム＋編集は，ヒューリスティクスによるルールが有効に作用しているかどうか比較して確認するため，辺の数をルールで抽出したものと同じ数にそろえ，ランダムに辺を接続したものを編集した．ランダム順序＋編集は，近い節点ペア順に編集を進めることの有効性を確認するために，順番をランダムにして編集を行うものと比較する．

評価指標は適合率（Precision）・再現率（Recall）・F 値（F-measure）を用いる．これらの評価指標は分類タスクにおいて標準的に用いられている評価指標で，類似のタスクの関係抽出でも利用されている [5, 6]．正解データの中で辺が接続している場合を正例，接続していない場合を負例と言い，正しく正例が予測できた数（True Positive，TP）・誤って正例と予測した数（False Positive，FP）・誤って負例と予測した数（False Negative，FN）の 3 値によってこれらの指標は

定義される．ぞれぞれの定義は以下のようになる．

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{4.2}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{4.3}$$

$$\text{F} - \text{measure} = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{4.4}$$

適合率はモデルの出力が正確にできているかどうかを示しており，モデルが誤って正例と予測するかどうかに着目した評価ができる．再現率は，正解のデータの正例の内，どれだけカバーして予測できたかどうかを評価可能である．F 値はそれらの調和平均を示しており，適合率と再現率のバランスを取った数値で，モデルの性能の指標となる．F 値は二種類の方法で計算し，辺のクラス毎に算出した F 値の平均を取るマクロ F 値と，クラスに関係なく全体の F 値を算出したマイクロ F 値を用いて評価する．マクロ F 値はうまく予測出来ていないクラスがあるような，クラスごとのばらつきを確認することができ，マイクロ F 値は全体的な予測性能が評価できる．適合率と再現率は，断りがない限りは，マイクロ F 値と同様に全体の数値から計算する．

4.2 ルールベースの抽出器

評価のためのベースラインと，グラフの編集モデルの編集対象のグラフを作成するためのルールベースのエッジ抽出器の詳細を示す．ルールベースの抽出器では，記述されたルールに従い，ルールに適合するものに対して辺を接続する．ルールは主に用語間の距離とその用語ペアのラベルによって定義する．本節では，合成手順コーパスの関係を抽出するためのルールを示す．ルールは例えば，接続しうる用語ラベルの用語のうち最も近いものに辺をつける，つまり材料の投入の辺であれば材料と操作間で文章中で最も近いものに辺をつける，といったような単純なルールとして定義する．

ルールの記述は, Head と Tail になる節点の用語ラベル毎に定義する. 辺の種類は Operation–Operation・Operation–Material・その他の辺の 3 種類に分類してルールを定義する．ルー

ルは節点ペアの用語ラベルごとに出現順序と距離に合わせて定義する．距離は用語ペア間に存在する語の数で定義され，文章をスペース区切りにすることで単語分割した．

4.2.1 Operation–Operation

Next_Operation：Operation–Operation の辺ラベルは，操作の進行を示す Next_Operation のみの一種類である．この辺は，操作の進行の順に Operation が記述されていると仮定して，Operation の用語が出現した順番に，前から後ろへ Next_Operation の辺をつける．

4.2.2 Operation–Material

Operation–Material 間の辺は，操作と材料間の関係を示すもので，辺ラベルは材料の投入を表す Recipe_Precursor・生成物の生成を示す Recipe_Target・中間生成物の生成を示す Participant_Material・操作の溶媒となる材料を示す Solvent_Material・操作の雰囲気を示す Atmospheric_Material の 5 クラスが存在する．これらはイベントとしてタグ付けされているが，ここでは変換して用語ペア間の辺に変換して解く．基本的には Material から文中で最も近傍に記述されている Operation に関係をつけるが，その時の関係クラスの判定基準を以下に示す．

Solvent_Material・Atmospheric_Material・Participant_Material：これらのクラスについては，クラス毎に辞書を用意し，マッチした辞書に割り当てられたクラスの辺を Material と最近傍の Operation に対する辺を接続する．Solvent_Material は溶媒・Atmospheric_Material は雰囲気・Participant_Material は中間材料となる材料である．これらの場合については特定の物質が該当する場合が多く，例えば，“water” は溶媒になり，“Ar” は雰囲気，“solution” のような一般的な語で表される Material は中間材料となる場合が多いといえる．そこで，溶媒や雰囲気になりやすいものをあらかじめ辞書として作成し，Tail となる Material がそれにマッチした場合はそのクラスに割り当てる．訓練データのうち，主に頻

出する上位の単語を辞書に追加するが，他のクラスとかぶっている場合や特徴的なものに関しては，抜いたり，正規表現で指定したりした．それぞれの辞書を Solvent_Material については表 11，Atmospheric_Material については表 12，Participant_Material については表 13に示した．

Recipe_Precursor・Recipe_Target：これらのクラスは，単純な辞書などでの分類が困難であり，文脈に基づいた判定が必要である．そこで，判定できない中でもより正しく予測するため，Solvent_Material，Atmospheric_Material，Participant_Material の辞書に該当しない Material をすべて，実際に訓練データに存在する辺の数がより多い Recipe_Precursor のクラスの辺として最近傍の Operation に対して接続する．つまり，このルールベースモデルでは Recipe_Target の辺は出力されない．

表11: Solvent_Material についての辞書

表現
.*water.*
.*(n|alcoh|glyc)ol.*
.*NaOH.*
.*HCl.*
.*acetone.*
.*acid.*
.*H2O.*
.*chloroform.*
.*sodium hydroxide.*
.*DMF.*
.*THF.*
.*N,N-dimethylformamide.*
.*hexane.*
.*toluene.*
.*H2SO4.*
.*EtOH.*

4.2.3 その他の関係

本節までに説明した関係以外のその他の関係ラベルは，材料に対する条件付けを示す Property_Of・操作に対する条件付けを示す Condition_Of・数値と単位の間の関係を示す Number_Of・材料に対する量の条件付けを示す Amount_Of・その他の条件付けを示す Descriptor_Of・数値条件に対する条件付けを示す Type_Of・材料や装置のブランドを示す Brand_Of・操作で利用する装置を示す Apparatus_Of・装置に対する数値条件付けを示す Apparatus_Attr_Of の 9 クラスある．これらのクラスに対しては，Head と Tail のクラスとその間の距離のみでルールを定義する．

Property_Of：この辺ラベルは，材料に対する数値条件などの条件付けを示す．辺は Property-Unit による数値条件付けとその他の条件付けの Property-Misc の 2 種類の用語ラベルを Head としてとる．Tail としては材料を示す Material と Nonrecipe-Material の 2 種類の用語ラベルをとる．この内，Head のラベル毎に場合分けしてルールを作成する．

Property-Unit を Head とする場合では，プロセスに対して直接関係する Material の

表12: Atmospheric_Material についての辞書

表現
.*air.*
.*argon.*
.*Ar.*
.*N2.*
.*nitrogen.*
.*H2.*
.*oxygen.*
.*O2.*
.*hydrogen.*
.*CH4.*
.*H2S.*
.*He.*

みに対する関係を考慮して，Property-Unit から文内でもっとも近傍に存在する Material に対して Property_Of の辺を割り当てる．これは，科学技術文献の特徴上，プロセスに関連しない材料に関しての数値条件を記述することは少なく，実際のルール開発データ内にも Property-Unit–Nonrecipe-Material の辺は 1 件のみしか存在しなかった．このことから，Property-Unit では，プロセスに関係する Material のみに対して関係を割り当てる．

Property-Misc を Head とする場合では，実際のデータ中に Material と Nonrecipe-Material の両者に対して関係を持つ場合が多く存在したため，文中で Property-Misc から最も近傍に存在する Material もしくは Nonrecipe-Material に対して接続する．Property-

表13: Participant_Material についての辞書

表現	
.*solution.*	.*precipitates.*
.*mixture.*	.*carbon.*
.*product.*	.*silica.*
.*samples.*	.*HCl.*
.*precipitate.*	.*NaBH4.*
.*powder.*	.*slurry.*
.*suspension.*	.*Cu.*
.*precursor.*	.*H2O.*
.*sample.*	.*Samples.*
.*products.*	.*ZnO.*
.*chemicals.*	.*KOH.*
.*powders.*	.*compound.*
.*GO.*	.*filtrate.*
.*solid.*	.*NaOH.*
.*pellets.*	.*films.*
.*solvent.*	.*graphene.*
.*materials.*	.*polymer.*
.*particles.*	.*CTAB.*
.*material.*	.*zeolite.*
.*gel.*	.*SnO2.*
.*reagents.*	.*membrane.*
.*precursors.*	.*hydrochloric aci.*

Misc–(Material|Nonrecipe-Material) の辺は，材料に対する様々な条件付けを付与する関係であるため，プロセスに関与する Material と関与しない Nonrecipe-Material のどちらに対する場合も多く存在しており，両者に対する関係を考慮して抽出する．近傍のペアで接続するのは，条件付けは係り受けを考慮すると，近隣に記述されることが多いためである．

Condition_Of：この辺ラベルは，温度条件のような，操作に対する条件付けを示すラベルで，数値条件を示す Condition-Unit とその他の条件である Condition-Misc を Head としてとる．この辺では全ての Head になりうる用語から文内で最も近傍に存在する Operation に対して，Condition_Of の関係を割り当てる．

Number_Of：この辺ラベルは，Head である数値 Number と Tail となる数値情報の単位 Property-Unit・Condition-Unit・Apparatus-Unit を結びつけるラベルである．文献中の数値は多くの場合で “100 K” や “1 cm2” のように数値の後ろに単位が記述されるケースが多い．このことから，Number から文内の後方で Property-Unit・Condition-Unit・Apparatus-Unit の内で最も近傍に記述されている用語に対して辺を接続する．

Amount_Of：この辺ラベルは, Head の量を表す単位の Amount-Unit とその他の量の条件 Amount-Misc から Tail となる材料の Material・Nonrecipe-Material へ接続して材料の量の条件を付加する．数値条件の量的単位の Amount-Unit とその他の量を示す条件 Amount-Unit から，プロセスに関与する材料 Material と関与しない材料 Nonrecipe-Material のうち，文内で最近傍のものに対して辺を接続する．

Descriptor_Of：この辺ラベルは，粉末状といったような材料の状態などの条件を示す Material-Descriptor や装置の状態といった条件 Apparatus-Descriptor を Head として，それぞれ対応する Tail となる材料の Material・Nonrecipe-Material と装置の Synthesis-Apparatus・Characterization-Apparatus に対する条件付けをする．

まず，材料の状態を示す Material-Descriptor から材料に対する関係では，文内で最も近傍に存在する Material 及び Nonrecipe-Material の両者に対して，Descriptor_Of の辺を接続する．また，装置の状態を示す Apparatus-Descriptor からの関係では，合成プロセ

スに関係する装置である Synthesis-Apparatus に対して辺を接続する．特性を調べる装置である Characterization-Apparatus は，合成プロセスに関係しないため，実際のデータ中でこの辺は小数しか存在していないことからここでは考慮しない．これは，プロセスに重きをおいてに記述されている文章では，特性を調べる装置については詳しい説明を必要としない場合が多いためだと考えられる．

Apparatus_Of：この辺ラベルは, 操作で利用する装置を指定するためのラベルで, Head として Synthesis-Apparatus と Characterization-Apparatus, Tail として Operation をとる．(Synthesis-Apparatus|Characterization-Apparatus)–Operation の辺では，“He side was detected with a gas chromatograph.” の様に, 操作に対する条件を後ろから修飾するように述べられる場合が多いため，Synthesis-Apparatus 及び Characterization-Apparatus から，前方で最も近傍に存在する Operation に対して辺を接続する．前方に存在しない場合は，後方に述べられている可能性を考え，後方で最も近傍に存在する Operation に対して辺を接続する．

Type_Of：この辺ラベルは，数値条件の種類などのタイプを示すためのラベルで，例えば，“heating rate of 2 degC min-1” のように，数値条件の種類を指定するラベルである．まず，材料の数値条件の種類についての Property-Type–Property-Unit と，装置の数値条件の種類を示す Apparatus-Property-Type–Apparatus-Unit については，それぞれ文中で最も近傍に存在する対応した単位に対して関係を割り当てる．操作の条件の種類についての Condition-Type–Condition-Unit については，ルール開発用データで抽出精度が高かったため，Condition-Type から文中の前方で最も近傍に存在する Condition-Unit に関係を割り当てる．

Brand_Of：この辺ラベルは，材料や装置のブランドを指定する辺である．材料のブランドについては，“hydrazine monohydrate (Alfa Aesar, 99%),” の様に，後ろに記述されることが多いため，Brand から前方に対し，文内で最も近傍に存在する Material もしくは Nonrecipe-Material に対して辺を接続する．装置のブランドについては，Brand から最近

傍の Synthesis-Apparatus・Characterization-Apparatus に対して辺を接続する．

Apparatus_Attr_Of：この辺ラベルは，“500 ml flask” のように，装置の属性となる値を示すためのラベルである．このような数値は近傍に記述されることが多いため，Apparatus-Unit から最近傍の Synthesis-Apparatus もしくは Characterization-Apparatus に対して辺を接続する．

Coref_Of：この辺ラベルは，同一材料を示すものをつなぐラベルである．同一の材料は文脈上やその語の構成から読み解かなければないため，ルールベースで解くことは困難であり，ほとんど正しい予測が出来ないため，ルールベースでは対象としない．

4.3 実験結果

編集モデルの有効性を確認するために，合成手順コーパスに対する抽出性能を評価する．合成手順コーパスに対する抽出手法は提案されていないため，作成した 4.2 節のルールベース手法で抽出したものをベースラインとして比較する．さらに，ルールベース手法の結果を編集することの有効性を確認するために，ルールベースで出力された辺を編集する場合（ルール＋編集）と，辺が全くついていない状態から編集を進める場合（編集のみ）に対して評価を行った．

表14: グラフの編集モデルとルールベース抽出器の比較

モデル		適合率	再現率	マイクロ F	マクロ F
ルールベース	開発	0.784	0.812	0.797	0.683
	評価	0.808	0.807	**0.807**	0.689
編集のみ	開発	0.819	0.883	0.850	0.815
	評価	0.769	0.823	0.795	0.720
ルール＋編集	開発	0.832	0.873	0.853	0.837
	評価	0.805	0.800	0.802	**0.788**
ランダム＋編集	開発	0.837	0.838	0.837	0.752
	評価	0.773	0.795	0.784	0.694
ランダム順序＋編集	開発	0.783	0.784	0.784	0.752
	評価	0.748	0.743	0.745	0.697

得られた結果は表 14に示した．まず，ルールベースと，提案手法であるルール＋編集を比較すると，開発データに対する性能では全てにおいて上昇しているが，評価データに対しての性能を比較すると，マイクロ F 値においては，ほぼ同値であるが，ルールベースのほうがわずかに高い性能を示している．一方で，マクロ F 値ではルール＋編集のほうが約 0.1 ポイントほど高い性能を示した．これらの結果から，全体的な性能はほぼ同等である一方で，クラスごとに性能を見ると，ルール＋編集のほうが高い性能を示していることから，それぞれのクラスで突出して予測できていないクラスはなく，バランスよく予測できていることを示している．

ルール＋編集と編集のみを比較すると，開発・評価ともにルール＋編集のほうが高い性能を示している．特にマクロ F 値についてはルール＋編集のほうが高い性能を示した．この結果から，ルールベースの抽出器の抽出結果を利用することで，クラスごとの性能のばらつきを減らすことができるといえる．

ヒューリスティクスに基づいたルールによって接続した辺の効果を確認するために，ルール＋編集と，辺をランダムに接続したランダム＋編集を比較すると，ルール＋編集のほうが高い性能が得られ，ヒューリスティクスの有効性が確認できた．ランダムな辺で接続した場合では，編集のみと比較すると，GCN によって節点同士で情報が伝播するが，ルール＋編集と比較すると，伝播する情報が規則性のないものになる．得られた結果は，ランダム＋編集は，ルール＋編集や編集のみよりもマイクロ F とマクロ F 両者において抽出性能が低くなった．これは，ランダムに伝播する情報がノイズとして作用してしまい，抽出性能が低下したと考えられる．この結果から，編集対象のグラフの辺は，ランダムではなく指向性がある接続をしている必要があると示唆される．

ランダム順序＋編集の抽出性能は，マイクロ F については一番低い性能で，マクロ F についてもルールからほとんど向上がみられなかった．ランダム順序＋編集は，編集する順序をランダムにして，その他はルール＋編集と同じ条件での実験である．この結果は，近い順に辺を編集する方策が有効であることを示しており，グラフの編集モデルで得られた性能に大きく寄与していることを示している．

4.3.1 クラスごとの評価

辺のクラスごとの抽出性能の評価を行う．この評価では，グラフの編集モデルが得意とするクラスを確認することで，モデルの傾向を評価できる．表 15はルールベースの抽出器における性能，表 16は辺が全く接続されていない状態から編集した編集のみにおける性能，表 17はルールベースモデルの結果を編集したルール＋編集の評価データに対する性能評価である．開発データに対する評価は付録 A に示した．これらを比較すると，編集モデルとルールベース抽出器は異なる出力をしていることがわかる．例えば，Next_Operation を比較すると，ルールベースの抽出器の出力を編集モデルに与えているにも関わらず，編集した後では性能が低くなっていることがわかる．一方で，Atmospheric_Material を比較すると編集によって大きく性能が上昇していることがわかる．このことから，編集モデルでは，文脈とグラフ上での整合性を考慮することで高い性能が得られている一方で，ルールによるヒューリスティクスが活用しきれていない部分がある

表15: 評価データに対するルールベース抽出器におけるクラス毎の抽出結果

辺クラス	適合率	再現率	F 値
Next_Operation	0.990	0.881	0.932
Recipe_Precursor	0.730	0.414	0.528
Recipe_Target	0.000	0.000	0.000
Participant_Material	0.419	0.800	0.550
Solvent_Material	0.697	0.418	0.522
Atmospheric_Material	1.000	0.378	0.549
Property_Of	0.905	1.000	0.950
Condition_Of	0.963	0.981	0.972
Number_Of	0.943	0.961	0.952
Amount_Of	0.744	0.865	0.800
Descriptor_Of	0.931	0.979	0.955
Type_Of	0.769	1.000	0.870
Brand_Of	0.561	0.920	0.697
Apparatus_Of	0.972	0.854	0.909
Apparatus_Attr_Of	0.909	0.769	0.833
Coref_Of	0.000	0.000	0.000

ことが示唆される．

編集のみとルール＋編集を比較すると，多くのクラスにおいてほとんど同等の性能が得られている一方で，Recipe_Target・Solvent_Material・Atmospheric_Material・Property_Of・Condition_Of・Type_Of・Apparatus_Attr_Of のクラスにおいてはルール＋編集の方が高い性能を得られている．ルール＋編集と編集のみの差はルールベースの抽出均の抽出結果を利用したことのみであることから，ルールベースを利用することでこの違いが生じたと考えられる．Property_Of・Condition_Of・Type_Of・Apparatus_Attr_Of については，ルールベースの抽出器によって高い性能での抽出が出来ているため，その抽出結果を継承することが出来たと考えられる．しかしながら，Recipe_Target・Solvent_Material・Atmospheric_Material については，ルールによる抽出では抽出がうまくできていない．要因として考えられるのは，ルール＋編集では，対象の節点が誤っているクラスであったとしても，入力の時点で辺で接続されているため，GCN によって周囲の情報が伝搬されて，クラス分類の性能

表16: 評価データに対する編集のみにおけるクラス毎の抽出結果

辺クラス	適合率	再現率	F 値
Next_Operation	0.772	0.768	0.770
Recipe_Precursor	0.494	0.710	0.583
Recipe_Target	0.545	0.750	0.632
Participant_Material	0.556	0.611	0.582
Solvent_Material	0.545	0.486	0.514
Atmospheric_Material	0.786	0.733	0.759
Property_Of	0.667	0.933	0.778
Condition_Of	0.822	0.898	0.859
Number_Of	0.990	0.972	0.981
Amount_Of	0.856	0.867	0.863
Descriptor_Of	0.980	0.943	0.962
Type_Of	0.308	1.000	0.471
Brand_Of	0.585	0.889	0.706
Apparatus_Of	0.639	0.767	0.697
Apparatus_Attr_Of	0.545	0.500	0.522
Coref_Of	0.786	0.917	0.846

が上がったと考えられる．これらのクラスは，周囲の情報を受けやすく，例えば Recipe_Target は最終的に生成される材料であることから，その材料の節点の周囲の情報を利用することで予測がしやすくなったと考えられる．Solvent_Material と Atmospheric_Material の場合ではそれぞれ溶媒と雰囲気であることから，条件として容積といったような情報がついている場合には，そのクラスを予測しやすくなる，といったような推論が可能になる．編集のみの場合では，元の用語が近い節点同士の予測時にはほとんどの辺が接続されておらず，徐々に辺が接続されていく様に編集が進む．そのため，辺がつきやすい近い節点ペアの予測時には周辺の情報が利用できないことから，抽出性能に差が出たと考えられる．

それぞれのクラスの性能について比較すると，まず Next_Operation では，ルールベースの方が F 値 0.932 と，編集された結果の F 値 0.778 と比較して大幅に高い性能を示していることがわかる．Next_Operation は文章中に対して記述された操作の順番に繋いでいく辺のラベルである．定義したルールは単純に出現した Operation の順番に Next_Operation を繋いでい

表17: 評価データに対するルール＋編集におけるクラス毎の抽出結果

辺クラス	適合率	再現率	F 値
Next_Operation	0.822	0.738	0.778
Recipe_Precursor	0.517	0.605	0.558
Recipe_Target	0.818	0.692	0.750
Participant_Material	0.524	0.619	0.568
Solvent_Material	0.788	0.473	0.591
Atmospheric_Material	0.929	0.867	0.897
Property_Of	0.905	0.864	0.884
Condition_Of	0.925	0.892	0.908
Number_Of	0.990	0.976	0.983
Amount_Of	0.769	0.869	0.816
Descriptor_Of	0.941	0.914	0.928
Type_Of	0.923	0.923	0.923
Brand_Of	0.634	0.897	0.743
Apparatus_Of	0.750	0.675	0.711
Apparatus_Attr_Of	0.909	0.667	0.769
Coref_Of	0.714	0.909	0.800

くという単純なものであったが，編集されたものより正確な抽出が出来ていた．これは，ほとんどの場合で本来出現パターンのみの情報で解くことができる Next_Operation の辺の抽出に対して，深層学習モデルを利用して文脈情報が入ることによってノイズとして作用してしまい，編集の性能を下げているということが予測できる．Apparatus_Of の辺についても同様に，出現パターンが重要であると考えられる．

一方で，その他の辺のラベルに対しては，編集したものの方が高い性能を示している．これは，その他の辺のラベルについては，付加した文脈情報が有効に作用し，さらにグラフ上での整合性を考慮したモデルが効果を発揮していると言える．例えばプロセスの出発材料と生成物を表す Recipe_Precursor と Recipe_Target では，主語や目的語になった Material と動詞となった Operation の係り受けのような情報に基づいた推論が重要となると予測できる．それぞれ溶媒と雰囲気を表す Solvent_Material や Atmospheric_Material では，対象となる材料の特性や構成元素・三態などの情報と，文脈から溶媒や雰囲気として利用されたのか，投入や生成された材料なのか判定する必要がある．Property_Of や Condition_Of などの条件付けも係り受けなどの文脈情報が重要であると言える．

このように，辺のクラスごとに必要な情報が異なり，利用した情報によって抽出した結果が異なることがわかる．編集するよりもルールベースの方が正しく抽出出来ている場合があるという点から，ルールベースの抽出器の結果を利用するか編集するかを選択的に定めるようにモデル化することでこの問題が解決できると予測でき，これは今後の課題とする．

4.3.2 モデルの編集の最大距離

モデルを編集するに当たって，編集を近い順番に行う最大距離 d_{max} の差について分析する．本来は，すべての距離において別々に編集を行うことで，すべての辺を近い順に抽出をするべきである．しかしながら，モデルを学習するためには計算の勾配を保存する必要があり，編集の回数が増えれば増えるほど勾配が増加し，計算コストも増加する．そこで，計算機のメモリ上で取りうる最大の編集回数となる $d_{max} = 15$ を最大として，距離ごとの性能を比較することで，d_{max}

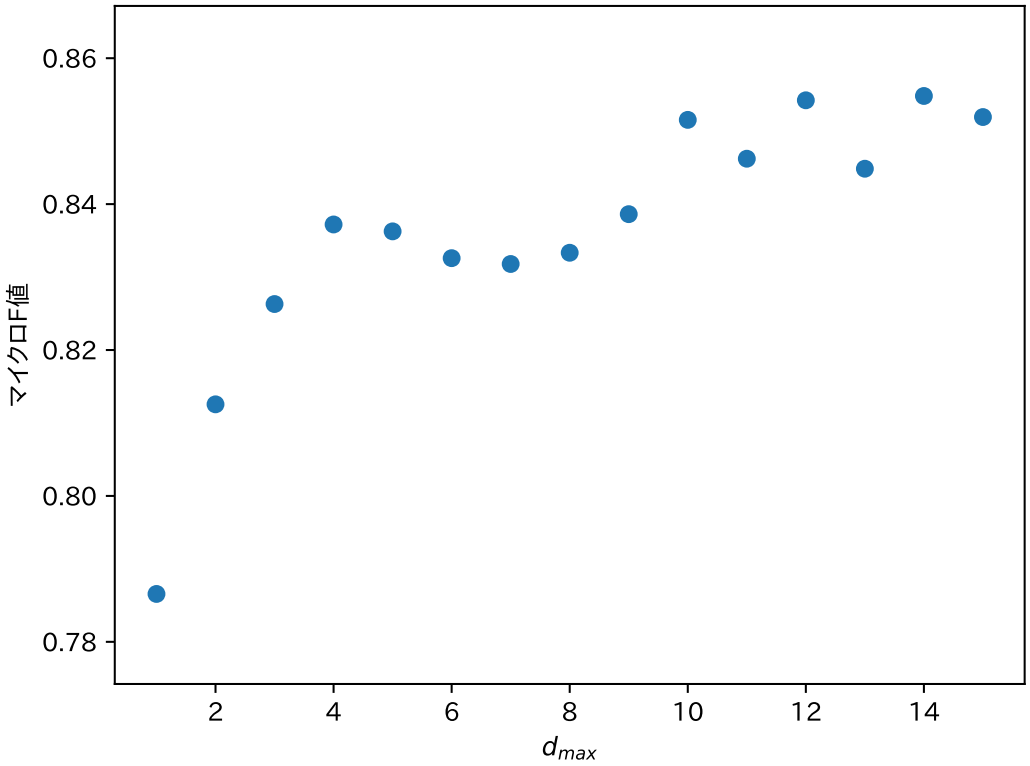


図7: 開発データに対する編集の最大距離 d_{max} とマイクロ F 値の関係

と性能の関係を確認する．実験は基本的に表 8と表 10に従ってモデルを構築するが，d_{max} のみ $1, 2, \cdots, 15$ と変化させて開発データに対するマイクロ F 値で評価する．

結果は図 7に示した．$d_{max} = 1$ はすべての辺を同時に定めている場合で，d_{max} が増加していくにつれて，距離が近いペア同士を独立した予測をして行くようになっていく．結果を見ると，可能な限り近いペア同士は独立させて予測させた方が高い性能が得られることがわかる．このことから，近い順に，簡単なものから難しいものへと編集を進める方策はグラフの抽出において効果的に作用することが示唆される．

また，d_{max} は，$d_{max} \geq 10$ で飽和していることが読み取れることから，$d_{max} \geq 10$ では同時に編集を行うことは問題ないということがこの結果から示せる．$d_{max} \to \infty$ として，すべての距離の辺を独立して編集するようにすると，計算コストが著しく増加するため困難である．しかし，本実験で $d_{max} \geq 10$ であればようことが示されたことから，ハイパパラメタチューニングで選択さ

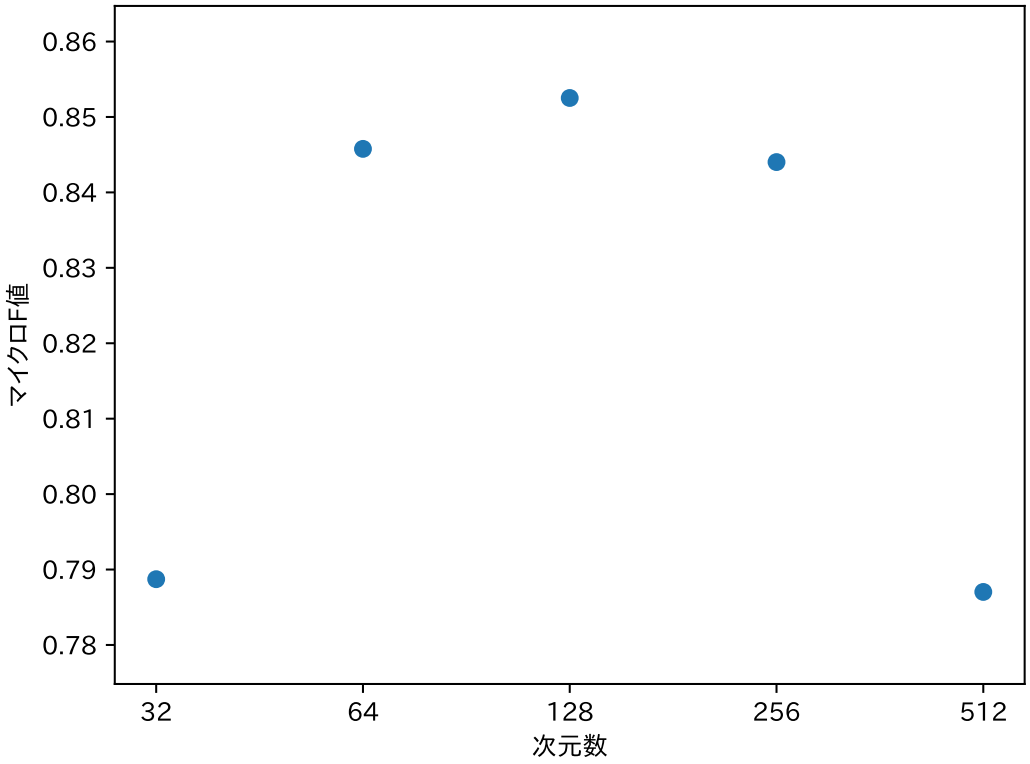


図8: グラフの編集モデルに対する次元数の特性

れた $d_{max} = 15$ で十分である．

4.3.3 パラメタの解析

チューニングして得られた表 10のパラメタと，チューニング時にその他のパラメタで学習した場合の性能と比較し，それぞれのパラメタに対して考察する．実験はチューニングで得られたパラメタについて，一つのパラメタのみを変化させて開発データに対するマイクロ F 値を確認することで，編集モデルに対するそれぞれのパラメタの特性を確認する．d_{max} については 4.3.2 節で行った実験がこれに該当するため，本節では省略する．

まず，隠れ層の次元数に対するグラフの編集モデルの特性を確認する．次元数を 2^n $(n = 5, 6, 7, 8, 9) = [32, 64, 128, 256, 512]$ の範囲として実験を行った．隠れ層の次元数を変化させた結

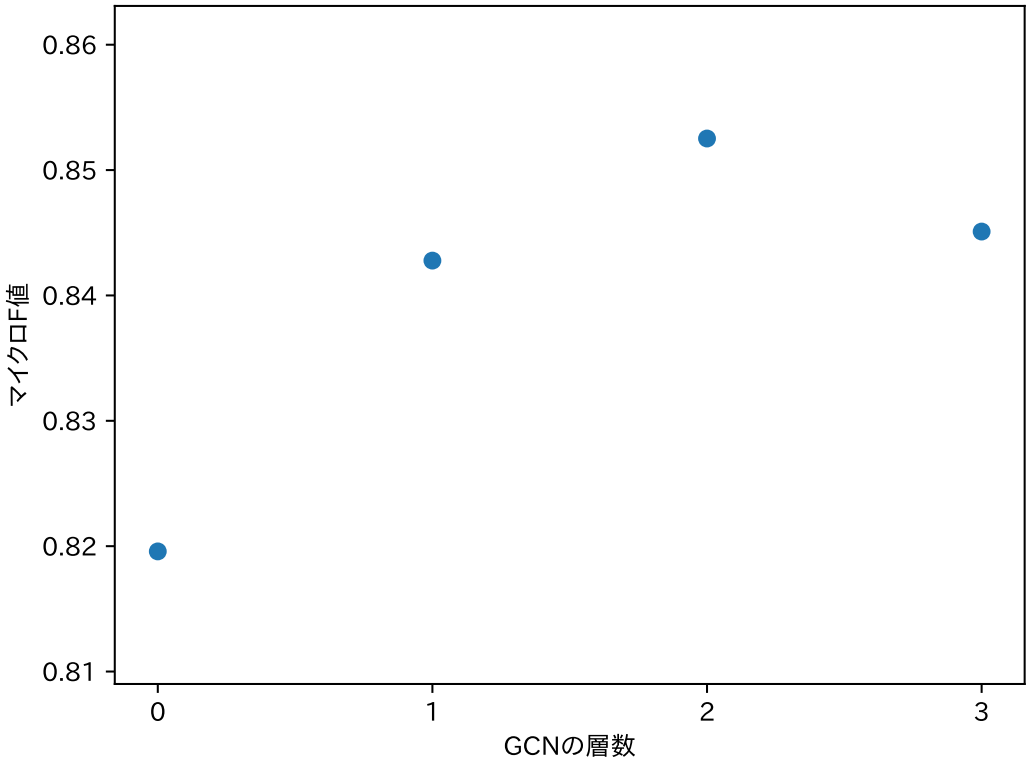


図9: グラフの編集モデルに対する GCN の層の数の特性

果を図 8に示した．結果を見ると，次元数が増えると性能が向上していき，最適なパラメタとして得られた 128 次元を最大にしてそれ以上大きくすると小さくなっていくことがわかる．このことから，128 次元が最適なパラメタとなっていることがわかる．

次元数を増加させると，モデル自身の表現力は，内部の重みの数が増えるため，表現力は向上する．しかしながら，図 8の結果から，単純に増加させるだけではなく，適切な数値を選択する必要があるとわかる．これは，モデルの表現力が高い場合では，訓練データに対して過適合してしまうためであると考えられる．過適合すると学習に利用されたデータに対しては正しい出力が可能だが，開発データや評価データのような未知のデータに対する性能が低くなる．これはパラメタ数が多いことによって，訓練データに出現するパターンを重みで記憶するように働いてしまい，汎化性能が下がるために生じる．このように過適合を生じさせないようにするために，次元数のような重みの数を設定するパラメタは適切に設定する必要がある．

次に GCN の層の数に対するグラフの編集モデルの性能を確認する．GCN の層の数は，グラフ上での考慮する隣接節点の最大距離を示している．例えば 0 層の GCN では自身の節点のみ，1 層では自身の節点と隣接節点，2 層では自身の節点と隣接節点に加えて隣接節点の隣接節点の情報を考慮した各節点の表現を計算する．本実験では考慮する節点の範囲に対して性能がどのように変化するか考察する．

実験は GCN を利用しない 0 層から 3 層までの範囲で行った．結果を見ると，GCN を利用しない場合では最低の性能を記録し，GCN の層の数を増加させると徐々に性能が向上していき，2 層で最大値をとり，3 層では性能が低下した．このことから，2 層が最適なパラメタとわかる．GCN では層の数を増やせば増やすほど考慮できる節点の範囲は増加するが，本実験では 3 層まで増加させると性能が低下した．これは，プロセス抽出において考慮する節点は GCN2 層で考慮可能な範囲で十分であったことを示唆しており，それ以上の距離の節点情報は冗長でノイズとして作用してしまっていると考えられる．

また，この結果から，グラフとしての特徴が有効であることが示せる．GCN が 0 層のとき GCN が利用されないため，グラフとしての特徴は利用せずに，文脈から得られた $\bar{\mathcal{N}}$ から直接予測予測するようになる．GCN を利用している 1 層以上の結果と比較すると，GCN を利用していない 0 層のマイクロ F 値は 0.02 ポイント以上低い結果が得られた．この結果から，情報抽出をグラフの抽出として定式化し，そのグラフとしての特徴を考慮して抽出すると効果的に抽出が可能であるといえる．

4.3.4 実際の事例

実際に抽出した例を比較して，その抽出の特性を確認する．グラフを抽出した内の一部を可視化したものを示す．対象としたのは図 10の文章で，図 11は正解データ，図 12はルールベース抽出器の出力，図 13はルールベース抽出器の出力を編集したものである．これらの図における節点と辺の色はそれぞれのクラスを表している．節点の語の後ろに「_」が記述されているものは，同じグラフ上に同一の語が含まれているために，それらを区別するために追加した．図 10内で下線が

10.1016/j.ssc.2007.04.044
Influence of Ni doping on the properties of perovskite molybdates SrMo1-xNixO3 (0.02<=x<=0.08)

A series of polycrystalline samples of SrMo1-xNixO4(0.02<=x<=0.08) were prepared through the conventional solid-state reaction method in air. Appropriate proportions of high-purity SrCO3, MoO3, and Ni powders were thoroughly mixed according to the desired stoichiometry, and then prefired at 900 [?]C for 24 h. The obtained powders were ground, pelletized, and calcined at 1000, 1100 and 1200 [?]C for 24 h with intermediate grinding twice. White compounds, SrMo1-xNixO4, were obtained. The compounds were ground and pressed into small pellets about 10 mm diameter and 2 mm thickness. These pellets were reduced in a H2/Ar (5%: 95%) flow at 920 [?]C for 12 h, and then the deep red colored products of SrMo1-xNixO3 were obtained.

図10: テキスト例 [4]

引かれている語句は節点となる用語である．

まず，ルールベースで抽出された結果と正解データを比較すると，全体的には正しく抽出できている部分も多いが，節点ペアの Head と Tail の組み合わせは正しいが辺のクラスの誤りがある場合や，辺が足りていない部分などの誤りが多く見られる．これは，文脈や材料ごとの特徴などの情報をあまり考慮していないルールでは材料がどの役割を持つ材料なのか十分に分類出来ないことが原因だと言える．特に右下部の「mixed」に着目すると，ルールでは「prepared」からの辺がつけられているが，実際には段落の最初に材料「SrMo1-xNixO4」を準備する，という宣言であるため，その後のプロセスに直接には関連しないため，「mixed」と「prepared」の間に辺は存在しない．このような問題は，文脈から判定しなければ判定が難しいため，ルールベースの抽出器で抽出するのは困難である．

ルール＋編集と正解を比較すると，一部の辺が抽出出来ていない場合が見られるが，ほとんど正解に近いグラフが抽出できている．中央少し上方の「pressed」から「reduced」への辺に着目してみると，ルールベースの抽出器では抽出できているにもかかわらず，ルール＋編集では抽出

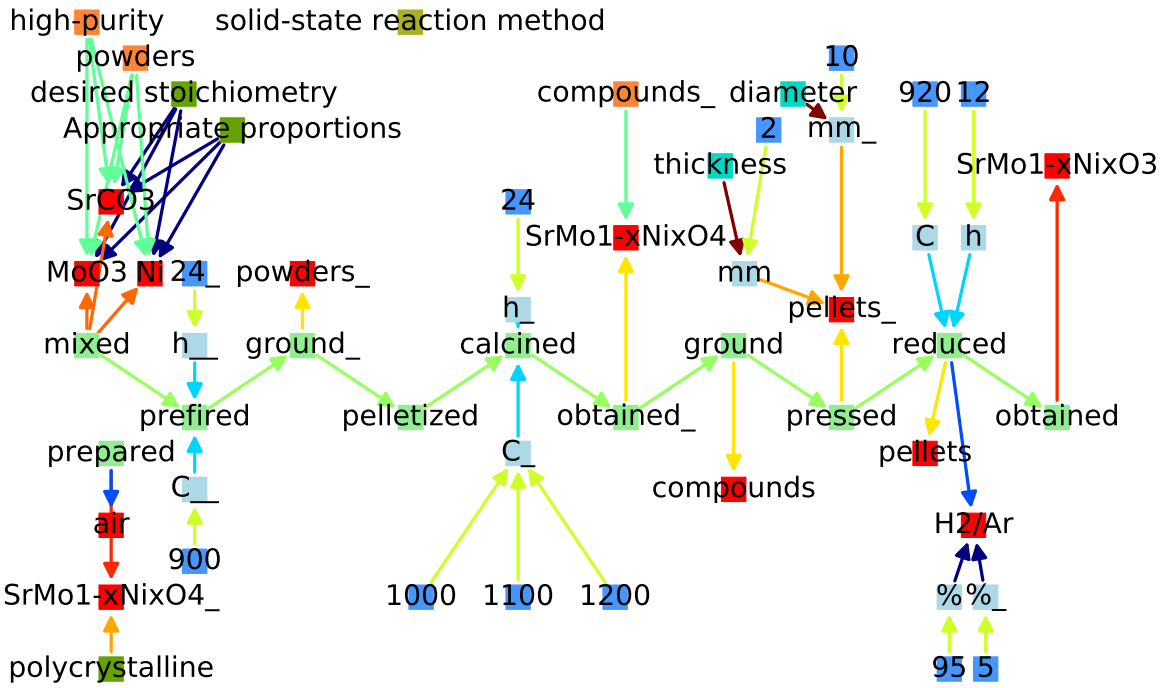


図11: 正解データのグラフ

できていないことがわかる．これは，深層学習モデルに対して入力されたことで，文脈のようなほかの情報が優先的に扱われてしまい，ルールによるヒューリスティクスがあまり考慮できていないことが原因だと考えられる．また，右下の「h__」と「C__」から「calcined」・中央少し右上の「C」と「h」「reduced」への辺が抽出できてない．もともとの文と見比べると，「calcined at 1000, 1100 and 1200 [?]C for 24 h」と「reduced in a H2/Ar (5%: 95%) flow at 920 [?]C for 12 h」はどちらも単純な文法で条件が記述されており，人間が見た場合では，専門家でなくても英語がある程度わかる人であれば抽出可能な内容である．類似のパターンでは，右下部の「h___」と「C___」から「preheated」の辺は抽出できている．「calcined」は焜焼する意味であり，意味的にも予測しやすい辺であったが予測できなかったが，「reduced」に関しては，「減らした」という意味の語であるため，温度条件や時間条件とは共起しにくいためであると考えられる．

このように，ルールベースの抽出器と深層学習による編集では，抽出内容に差異があるとわかる．本提案手法はルールベース抽出器の出力を編集するモデルによってルールベース抽出器と深

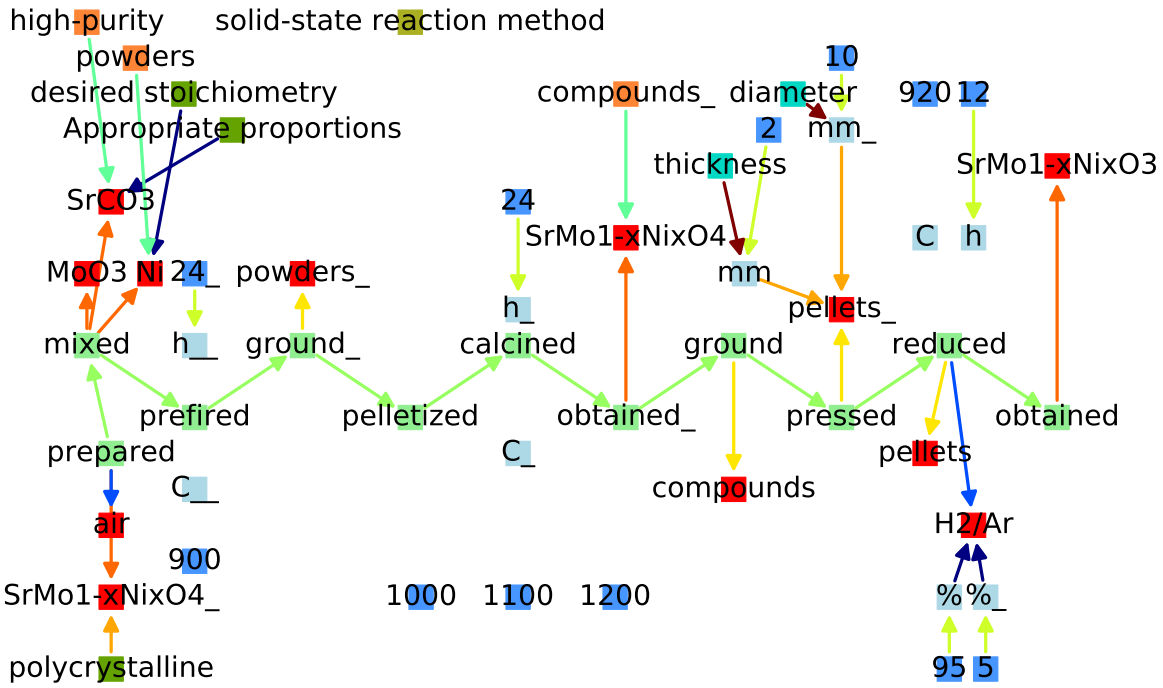


図12: ルールベース抽出器の出力グラフ

層学習器の利点を活かすことを目的に編集を行った．しかしながら，実際には後段にある深層学習モデルによって，ルールによるヒューリスティクスの特徴があまり利用できていない抽出結果が得られた．今後は編集前のグラフと編集後のグラフの両者の利点を活かすことが可能な組み合わせ方を模索する必要がある．

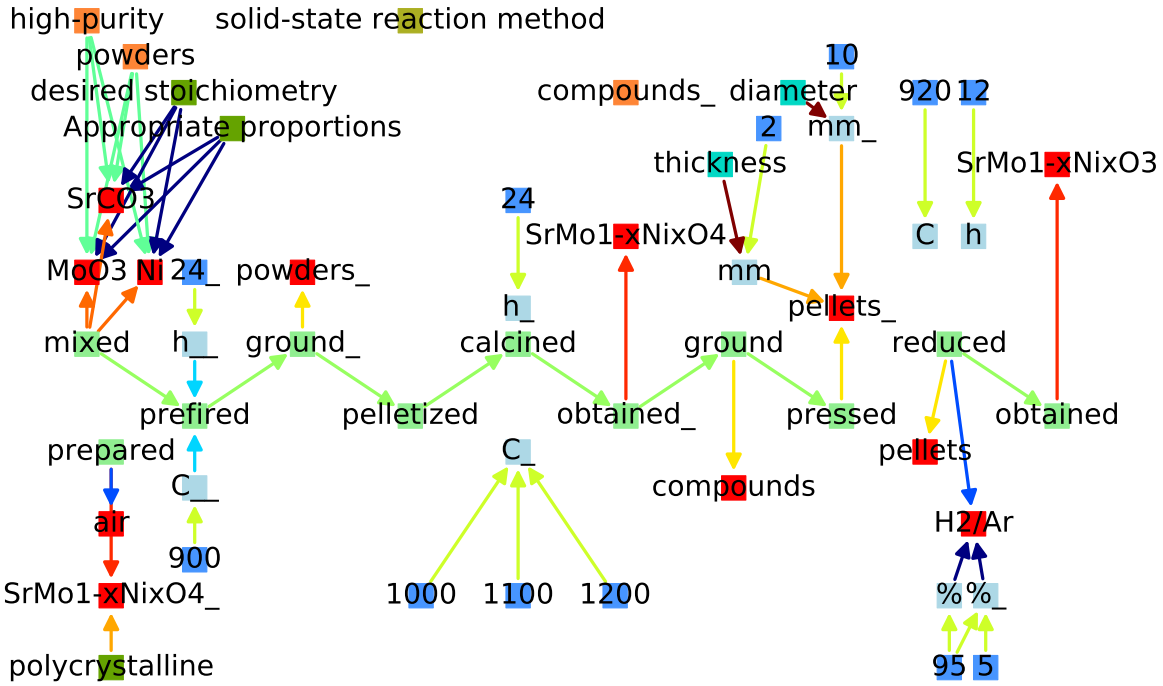


図13: ルール＋編集の出力グラフ

5. 結論

本論文をまとめるに当たって，本論文のまとめと今後の課題について述べる．

5.1 まとめ

本研究では，文献からの情報抽出をグラフの抽出として定式化し，その抽出を利用性が高くなるよう，グラフの編集によってグラフの抽出に取り組み，そのための一歩として，辺の編集に取り組んだ．グラフ構造は様々な形式の情報を網羅的に表現可能な形式であることから，適用範囲が広いと考えられ，本研究ではその中でも需要の高い無機材料分野の合成プロセス抽出を対象として有効性を確認した．合成プロセスの抽出では最先端の結果が得られ，ベースラインとしたルールで抽出したグラフと比較して，ルールで抽出したグラフを編集した場合ではマクロ F 値で約 0.1 ポイントの向上が見られた．一方で，ルールを利用せずに，辺が全く接続されていない節点のみの状態のグラフを編集することで徐々に辺を接続した場合でも，ある程度の性能が得られることを示した．

グラフの編集モデルでは，編集のために深層学習モデルのグラフの辺分類モデルを利用した．文脈上・グラフ上での整合性がとれた辺の分類を行うために，各節点の初期ベクトルとして Longformer のような Transformer ベースの事前学習モデルによって得られた特徴を利用し，GCN によって辺に沿って節点のベクトルを伝搬させることで，文脈とグラフ構造の両者を考慮した表現を獲得し，この表現に基づいて辺を分類した．編集はすべての節点ペアの間の辺を分類して，その結果によって上書きすることで行った．この時の順序を，元の文章中で隣接している節点ペア間の辺から順番に，徐々に離れているペア間の辺を分類するようにすることで，簡単な問題から難しい問題へと解き進めるように作用し，性能が向上することを示した．

5.2 今後の課題

本研究を経て必要となる今後の課題について述べる．

- 編集前のグラフの知識をより利用しやすいモデルの作成

  クラスごとに抽出性能を比較すると，編集モデルの入力として利用されているルールによる抽出結果のほうが高い性能を示しているクラスがあり，深層学習モデルに入力して得られた出力で上書きしたため，ルールに利用したヒューリスティクスが利用しきれていない．グラフの編集モデルでは既存のシステムによって抽出された情報を編集するため，既存のシステムとグラフの編集モデルの二つのモデルでの抽出をする．編集前のグラフとグラフの編集モデルによる抽出の両者のうち正しいものを選択すればより正確に抽出が可能になるが，本研究で提案したグラフの編集モデルではグラフの編集モデルの出力が優先され，編集前のグラフの情報はあまり利用できていなかった．

  この問題を解決するため，編集前のグラフとグラフの編集モデルの出力のうち，正しい辺を取捨選択するような機構を作成する必要があるといえる．提案したグラフの編集モデルは，編集前のグラフを入力としたグラフの辺の分類モデルによって抽出された結果によって編集前のグラフに関係なく上書きすることで編集を実現している．そのため，グラフの辺の分類モデルに利用されている深層学習モデルの特性が出やすい．辺の分類モデルでは，編集前の辺の情報を直接的には利用しておらず，GCN によってグラフの構造情報としてしか利用していない．そのため，入力された辺がどのような辺であったかはあまり考慮できていない．編集前のグラフの知識を利用しやすいように，編集前のグラフの情報を取捨選択して導入するモデルの作成が必要である．

- グラフの編集モデルの拡張

  本研究で提案したグラフの編集モデルは，辺の編集に対する提案のみで，節点の編集については実現できていない．そのため，辺の編集のみは可能であるが，任意のグラフの編集や，

節点が与えられていない状態での情報抽出は現状ではできないモデルとなっている．節点の編集は，既存の辺に直接的に影響を及ぼすため困難であり，例えば辺が接続されている節点を削除しようとすると，その辺は片方の節点のみにしか接続していない辺になり，不整合が生じる，といった事象が起こる．

この問題点を解決するために，グラフの編集モデルを節点の編集が可能なモデルに拡張する必要がある．実際には，単語や語句といった最小要素まで分割したものを節点として，その節点を用語としてグループ化し，そのグループ同士で辺を接続することで実現する方法が考えられる．

- 編集前のグラフの抽出

  本研究での提案では，編集前のグラフはルールベースの抽出器を用いているが，ルールを作成するのは人手で行う必要があり，そのコストが高いという問題点がある．この問題点を解決するため，ルールベースの抽出器の作成を自動化する手法や，機械学習などのデータ駆動の手法などを用いることで，このコストを削減する必要がある．

  一方で，編集前のグラフを全く辺がついてない状態から編集して辺を接続した編集のみでもある程度同程度の性能が得られることが本研究で示された．そこで，ほかのアプローチとして，編集前のグラフは抽出を目的としたものではなく，例えば節点を出現順に接続する，既存の手法による構文木を初期グラフとする，などのように，編集前のグラフを，コストが低い方法で変更して性能の向上を目指す方法がある．

- その他の情報抽出に対する有効性の確認

  グラフの編集モデルはグラフ構造の抽出として変換可能な問題であれば適用可能な手法であるため，様々な情報抽出タスクに応用が可能なモデルだと予測できるが，検証できていない．本研究では，無機材料分野の文献に対する合成プロセス情報の抽出を対象として取り組んだが，その他の情報抽出タスクにも応用が可能なモデルである．そこで，様々なケースに対して適用して，グラフの編集モデルの適用範囲や有効性を検証する必要がある．

謝辞

本論文は修士研究の成果をまとめたものです．本研究を行うに当たって多くの方にお世話になりました．特に本研究を行うにあたり，ご指導を賜った佐々木裕教授及び三輪誠准教授にお世話になりました．研究テーマを修士一年次の途中で本研究のテーマへと変更することになり，多くの場合が修士研究が課題研究の延長線上となるのに対して，スケジュールが短い状態でした．このような中で，本研究がまとめられたのは，偏に佐々木裕教授・三輪誠准教授のご指導の賜物であり，心より感謝申し上げます．

また，学部四年次に所属していた知能情報メディア研究室の浮田宗伯教授には，現在の研究の礎となる技術や知識をご指導いただきました．適用範囲はコンピュータビジョンと自然言語処理分野と異なりましたが，機械学習技術の基礎となる内容や考え方をご教示いただき，これが本研究をまとめるにあたっての基盤となりました．知能数理研究室に所属を移した後にも副指導教員としてお世話になり，気にかけていただきました．感謝申し上げます．

知能数理研究室のメンバーには，研究に関する議論やアドバイスをしていただき，その中から着想を得ることもあり，助けられることが多く有りました．博士学生である辻村有輝さん，浅田真生さんには技術や知識などの部分でご教示いただき，助けていただきました．同級生の方々とは研究内容の議論だけでなく，世間話など，日常的なことをともに行い，研究室での生活で楽しく過ごすことができました．感謝申し上げます．

最後に家族には，研究を進めるうえで必要である，金銭面や生活面で助けていただきました．特に金銭面では，生活に必要なものを購入していただいたり，学費の援助をしていただくなど，多くの部分で助けていただきました．生活面では，食料の仕送りや，生活のためのアドバイスなどをしていただき，研究に打ち込めるように支持していただきました．感謝申し上げます．

参考文献

[1] Sheshera Mysore, Zachary Jensen, Edward Kim, Kevin Huang, Haw-Shiuan Chang, Emma Strubell, Jeffrey Flanigan, Andrew McCallum, and Elsa Olivetti. The materials science procedural text corpus: Annotating materials synthesis procedures with shallow semantic structures. In *Proceedings of the 13th Linguistic Annotation Workshop*, pages 56–64, Florence, Italy, August 2019. Association for Computational Linguistics.

[2] Meiri Wang, Zhang Yining, and H. Zhang. A microsized cagelike sulfur/carbon composite for a lithium/sulfur battery with excellent performance. *ChemPlusChem*, 79:919–924, 2014.

[3] Iz Beltagy, Matthew E. Peters, and Arman Cohan. Longformer: The long-document transformer. *arXiv:2004.05150*, 2020.

[4] S.B. Zhang, Y.P. Sun, B.C. Zhao, X.B. Zhu, and W.H. Song. Influence of ni doping on the properties of perovskite molybdates srmo1 − xnixo3Â (0.02≤x≤0.08). *Solid State Communications*, 142(12):671 – 675, 2007.

[5] Makoto Miwa and Mohit Bansal. End-to-end relation extraction using LSTMs on sequences and tree structures. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1105–1116, Berlin, Germany, August 2016. Association for Computational Linguistics.

[6] Guoshun Nan, Zhijiang Guo, Ivan Sekulic, and Wei Lu. Reasoning with latent structure refinement for document-level relation extraction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1546–1557, Online, July 2020. Association for Computational Linguistics.

[7] Livio Baldini Soares, Nicholas FitzGerald, Jeffrey Ling, and Tom Kwiatkowski. Match-

ing the blanks: Distributional similarity for relation learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2895–2905, Florence, Italy, July 2019. Association for Computational Linguistics.

[8] Erik F. Tjong Kim Sang and Fien De Meulder. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, pages 142–147, 2003.

[9] Yuhao Zhang, Victor Zhong, Danqi Chen, Gabor Angeli, and Christopher D. Manning. Position-aware attention and supervised data improve slot filling. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP 2017)*, pages 35–45, 2017.

[10] Yuan Yao, Deming Ye, Peng Li, Xu Han, Yankai Lin, Zhenghao Liu, Zhiyuan Liu, Lixin Huang, Jie Zhou, and Maosong Sun. DocRED: A large-scale document-level relation extraction dataset. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 764–777, Florence, Italy, July 2019. Association for Computational Linguistics.

[11] Olga Kononova, Haoyan Huo, Tanjin He, Ziqin Rong, Tiago Botari, Wenhao Sun, Vahe Tshitoyan, and Gerbrand Ceder. Text-mined dataset of inorganic materials synthesis recipes. *Scientific Data*, 6, 12 2019.

[12] C. Walker and Linguistic Data Consortium. *ACE 2005 Multilingual Training Corpus*. LDC corpora. Linguistic Data Consortium, 2005.

[13] David Wadden, Ulme Wennberg, Yi Luan, and Hannaneh Hajishirzi. Entity, relation, and event extraction with contextualized span representations. 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)*, pages 5784–5789, Hong Kong, China, November 2019. Association for Computational Linguis-

tics.

[14] Pankaj Gupta, Hinrich Schütze, and Bernt Andrassy. Table filling multi-task recurrent neural network for joint entity and relation extraction. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 2537–2547, Osaka, Japan, December 2016. The COLING 2016 Organizing Committee.

[15] Alan Akbik, Duncan Blythe, and Roland Vollgraf. Contextual string embeddings for sequence labeling. In *COLING 2018, 27th International Conference on Computational Linguistics*, pages 1638–1649, 2018.

[16] Peng Lu, Ting Bai, and Philippe Langlais. SC-LSTM: Learning task-specific representations in multi-task learning for sequence labeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2396–2406, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[17] Jin-Dong Kim, Tomoko Ohta, Yuka Tateisi, and Jun'ichi Tsujii. Genia corpus—a semantically annotated corpus for bio-textmining. *Bioinformatics (Oxford, England)*, 19 Suppl 1:i180–2, 02 2003.

[18] Fusataka Kuniyoshi, Kohei Makino, Jun Ozawa, and Makoto Miwa. Annotating and extracting synthesis process of all-solid-state batteries from scientific literature. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 1941–1950, Marseille, France, May 2020. European Language Resources Association.

[19] Patrick Verga, Emma Strubell, and Andrew McCallum. Simultaneously self-attending to all mentions for full-abstract biological relation extraction. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 872–884, New Orleans, Louisiana, June 2018. Association for Computational Linguistics.

[20] Jiao Li, Yueping Sun, Robin Johnson, Daniela Sciaky, Chih-Hsuan Wei, Robert Leaman, Allan Peter Davis, Carolyn Mattingly, Thomas Wiegers, and Zhiyong lu. Biocreative v cdr task corpus: a resource for chemical disease relation extraction. *Database*, 2016:baw068, 05 2016.

[21] Allan Peter Davis, Cynthia G. Murphy, Cynthia A. Saraceni-Richards, Michael C. Rosenstein, Thomas C. Wiegers, and Carolyn J. Mattingly. Comparative Toxicogenomics Database: a knowledgebase and discovery tool for chemicalâ geneâ disease networks. *Nucleic Acids Research*, 37(suppl_1):D786–D792, 09 2008.

[22] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[23] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations (ICLR)*, 2017.

[24] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc., 2017.

[25] Hai-Long Trieu, Thy Thy Tran, Khoa N A Duong, Anh Nguyen, Makoto Miwa, and Sophia Ananiadou. DeepEventMine: End-to-end Neural Nested Event Extraction from Biomedical Texts. *Bioinformatics*, 06 2020. btaa540.

[26] Jian Liu, Yubo Chen, Kang Liu, Wei Bi, and Xiaojiang Liu. Event extraction as machine reading comprehension. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1641–1651, Online, November 2020. Association for Computational Linguistics.

[27] Jermsak Jermsurawong and Nizar Habash. Predicting the structure of cooking recipes. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 781–786, Lisbon, Portugal, September 2015. Association for Computational Linguistics.

[28] Mario Bollini, Stefanie Tellex, Tyler Thompson, Nicholas Roy, and Daniela Rus. Interpreting and executing recipes with a cooking robot. In *Experimental Robotics*, pages 481–495. Springer, 2013.

[29] Chaitanya Kulkarni, Wei Xu, Alan Ritter, and Raghu Machiraju. An annotated corpus for machine reading of instructions in wet lab protocols. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 97–106, New Orleans, Louisiana, June 2018. Association for Computational Linguistics.

[30] Maxwell Bates, Aaron J. Berliner, Joe Lachoff, Paul R. Jaschke, and Eli S. Groban. Wet lab accelerator: A web-based application democratizing laboratory automation for synthetic biology. *ACS Synthetic Biology*, 6(1):167–171, 2017. PMID: 27529358.

[31] Iris Hendrickx, Su Nam Kim, Zornitsa Kozareva, Preslav Nakov, Diarmuid Ó Séaghdha, Sebastian Padó, Marco Pennacchiotti, Lorenza Romano, and Stan Szpakowicz. SemEval-2010 task 8: Multi-way classification of semantic relations between pairs of nominals. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 33–38, Uppsala, Sweden, July 2010. Association for Computational Linguistics.

[32] Mike Mintz, Steven Bills, Rion Snow, and Daniel Jurafsky. Distant supervision for

relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 1003–1011, Suntec, Singapore, August 2009. Association for Computational Linguistics.

[33] Fenia Christopoulou, Makoto Miwa, and Sophia Ananiadou. Connecting the dots: Document-level neural relation extraction with edge-oriented graphs. 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)*, pages 4925–4936, Hong Kong, China, November 2019. Association for Computational Linguistics.

[34] Tsu-Jui Fu, Peng-Hsuan Li, and Wei-Yun Ma. GraphRel: Modeling text as relational graphs for joint entity and relation extraction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1409–1418, Florence, Italy, July 2019. Association for Computational Linguistics.

[35] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25, pages 1097–1105. Curran Associates, Inc., 2012.

[36] F. Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, pages 65–386, 1958.

[37] Bing Xu, Naiyan Wang, Tianqi Chen, and Mu Li. Empirical evaluation of rectified activations in convolutional network, 2015.

[38] Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. On the importance of initialization and momentum in deep learning. In Sanjoy Dasgupta and David McAllester, editors, *Proceedings of the 30th International Conference on Machine Learn-*

ing, volume 28 of *Proceedings of Machine Learning Research*, pages 1139–1147, Atlanta, Georgia, USA, 17–19 Jun 2013. PMLR.

[39] Alex Graves. Generating sequences with recurrent neural networks. *CoRR*, abs/1308.0850, 2013.

[40] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[41] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[42] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

[43] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay

Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[44] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014.

[45] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[46] Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana, June 2018. Association for Computational Linguistics.

[47] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach, 2019.

[48] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. Albert: A lite bert for self-supervised learning of language representations. In *International Conference on Learning Representations*, 2020.

[49] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735â 1780, November 1997.

[50] Mike Schuster and Kuldip Paliwal. Bidirectional recurrent neural networks. *Signal Processing, IEEE Transactions on*, 45:2673 – 2681, 12 1997.

[51] Niki Parmar, Ashish Vaswani, Jakob Uszkoreit, Lukasz Kaiser, Noam Shazeer, Alexander Ku, and Dustin Tran. Image transformer. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 4055–4064, StockholmsmÃ¤ssan, Stockholm Sweden, 10–15 Jul 2018. PMLR.

[52] Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. In *ICLR*, 2016.

[53] Makoto Miwa and Yutaka Sasaki. Modeling joint entity and relation extraction with table representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1858–1869, Doha, Qatar, October 2014. Association for Computational Linguistics.

[54] Shuai Ma, Gang Wang, Yansong Feng, and Jinpeng Huai. Easy first relation extraction with information redundancy. 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)*, pages 3851–3861, Hong Kong, China, November 2019. Association for Computational Linguistics.

[55] Matthias Fey and Jan E. Lenssen. Fast graph representation learning with PyTorch Geometric. In *ICLR Workshop on Representation Learning on Graphs and Manifolds*, 2019.

[56] Mark Neumann, Daniel King, Iz Beltagy, and Waleed Ammar. ScispaCy: Fast and Robust Models for Biomedical Natural Language Processing. In *Proceedings of the 18th BioNLP Workshop and Shared Task*, pages 319–327, Florence, Italy, August 2019. Association for Computational Linguistics.

付録

A. 開発データにおけるクラスごとの結果

開発データに対するクラスごとに見た性能を示す．クラスごとの評価は 4.3.1 節で行ったが，評価データに基づいた評価のみしか行っていない．本章では開発データに対する性能を示す．表 18はルールによる抽出，表 19はルールベースの抽出器の出力を編集したルール＋編集，表 20は辺が全くついていない状態から編集で辺をつけた編集のみの開発データに対する結果である．

表18: ルールベース抽出器におけるクラス毎の抽出結果

辺クラス	適合率	再現率	F 値
Next_Operation	0.995	0.929	0.961
Recipe_Precursor	0.716	0.338	0.459
Recipe_Target	0.000	0.000	0.000
Participant_Material	0.487	0.833	0.615
Solvent_Material	0.643	0.535	0.590
Atmospheric_Material	0.909	0.303	0.455
Property_Of	0.771	0.931	0.844
Condition_Of	0.917	0.953	0.934
Number_Of	0.927	0.962	0.944
Amount_Of	0.685	0.809	0.742
Descriptor_Of	0.791	0.935	0.857
Type_Of	0.714	1.000	0.833
Brand_Of	0.643	0.871	0.740
Apparatus_Of	1.000	0.909	0.952
Apparatus_Attr_Of	1.000	1.000	1.000
Coref_Of	0.000	0.000	0.000

表19: ルール＋編集におけるクラス毎の抽出結果

辺クラス	適合率	再現率	F 値
Next_Operation	0.804	0.851	0.827
Recipe_Precursor	0.672	0.703	0.687
Recipe_Target	0.806	0.694	0.746
Participant_Material	0.655	0.733	0.692
Solvent_Material	0.714	0.741	0.727
Atmospheric_Material	0.909	0.714	0.800
Property_Of	0.857	0.833	0.845
Condition_Of	0.924	0.938	0.931
Number_Of	0.972	0.982	0.977
Amount_Of	0.892	0.866	0.879
Descriptor_Of	0.846	0.865	0.856
Type_Of	0.571	1.000	0.727
Brand_Of	0.667	0.933	0.778
Apparatus_Of	0.950	0.905	0.927
Apparatus_Attr_Of	1.000	1.000	1.000
Coref_Of	0.917	0.846	0.880

表20: 編集のみにおけるクラス毎の抽出結果

辺クラス	適合率	再現率	F 値
Next_Operation	0.837	0.885	0.860
Recipe_Precursor	0.687	0.692	0.691
Recipe_Target	0.484	0.833	0.612
Participant_Material	0.628	0.763	0.689
Solvent_Material	0.821	0.697	0.754
Atmospheric_Material	0.818	0.900	0.857
Property_Of	0.800	0.875	0.836
Condition_Of	0.864	0.905	0.884
Number_Of	0.963	0.991	0.977
Amount_Of	0.885	0.833	0.858
Descriptor_Of	0.846	0.906	0.875
Type_Of	0.571	0.571	0.571
Brand_Of	0.667	0.933	0.778
Apparatus_Of	0.850	0.944	0.900
Apparatus_Attr_Of	1.000	1.000	1.000
Coref_Of	0.917	1.000	0.957

B. ハイパパラメタチューニング

4.1.3 節で述べたハイパパラメタチューニングで得られた結果について述べる．表 21はチューニングにおいて得られたすべての結果である．

表21: ハイパパラメタチューニングの結果

ID	GCN の層の数	次元数	d_{max}	開発データに対するマイクロ F 値
1	0	64	1	0.753
2	0	64	5	0.808
3	0	64	10	0.821
4	0	64	15	0.816
5	0	128	1	0.774
6	0	128	5	0.799
7	0	128	15	0.82
8	0	256	1	0.78
9	0	256	5	0.812
10	0	256	10	0.815
11	0	256	15	0.83
12	1	64	1	0.786
13	1	64	5	0.845
14	1	64	10	0.839
15	1	64	15	0.824
16	1	128	1	0.799
17	1	128	5	0.833

18	1	128	10	0.837
19	1	128	15	0.843
20	1	256	1	0.805
21	1	256	5	0.836
22	1	256	10	0.847
23	1	256	15	0.844
24	2	64	1	0.786
25	2	64	5	0.82
26	2	64	10	0.846
27	2	64	15	0.846
28	2	128	1	0.802
29	2	128	5	0.834
30	2	128	10	0.847
31	2	128	15	0.853
32	2	256	1	0.798
33	2	256	5	0.828
34	2	256	10	0.841
35	2	256	15	0.844
36	3	64	1	0.789
37	3	64	5	0.829
38	3	64	10	0.851
39	3	64	15	0.847
40	3	128	1	0.794
41	3	128	5	0.835
42	3	128	10	0.848

43	3	128	15	0.845
44	3	256	1	0.788
45	3	256	5	0.83
46	3	256	10	0.845
47	3	256	15	0.841
48	0	128	10	0.811