Transformer を用いた非構造化データの構造化

Structuring Non-Structured Data by Transformer

川崎拳人 [1]

Kento Kawasaki[1]

[1]株式会社リーディング・エッジ社　研究開発部

[1]Leading Edge Co.,Ltd Research and Development Department.

Abstract: It's important to smoothly collect and manage high quality data in the society where demand is growing for artificial intelligence and data science. However, many of them are supported by structured data. Non-structured data such as document and e-mail are not making full use of in those areas because it is difficult to collect and manage non-structured data.

Structuring non-structured data needs high ability like a human which read all of document, understand useful information and deem identically various expression which has the same meaning. This paper adopted Transformer to solve problems as mentioned above.

Transformer is the basis of BERT and GPT-2 and often used in machine translation. This paper applied Transformer to Structuring non-structured data. Namely, Translating non-structured data into structured data. In particular, this paper researched method of information extraction, notation conversion and unifying expression. Transformer has attention mechanism. Firstly, Self Attention encode relationship between words in the sentence and useful for select important information in sentence. Secondly, Encoder-Decoder Attention find relationships between words in source sentence and words in target sentence and useful for integrating various notations and expression.

This paper evaluated the method with Exact Match and F-value. Those are over 90. This paper also considered attention weight and discovered the effectiveness of Transformer on the task of information extraction and notation conversion.

However, there are some tasks on unifying expression of non-structured data, it needs BERT model which can interpret higher context. And although this paper structured non-structured data as translation task, it had side effects of it. So we want to strive to research while including information extraction by machine comprehension.

1. はじめに

今日の AI・データサイエンスに対する社会需要の高まりの中で，円滑に質の高いデータを収集，管理することは重要になっている．しかしながら，その多くはデータベースなどの構造化データに限られ，文書やメールなどの非構造化データは，収集や管理の難しさから，AI・データサイエンスの現場で生かしきれていない．2017 年に総務省が企業を対象に調査した「安心・安全なデータ流通・利活用に関する調査研究」によると，「産業データの取り扱いや利活用において，現在または今後想定される課題や障壁」として約 50%の企業が「データの収集・管理に価格コストの増大（データのフォーマット等が共通化されていない，データ品質の管理等）」を挙げており，列挙された障壁の中で一番高い結果となった[1]．また，アメリカ，イギリス，ドイツにおいても同様の結果が出ており，非構造化データの活用に対する期待が高いものの，具体的な活用方法は見出されていないと考えられる．

本稿では，非構造化データとして，求人情報を含むメール文書を対象とし，BERT や GPT-2 など自然言語処理分野における先端技術のベースとなっている Transformer を用いた構造化手法を紹介する．また，非構造化データの構造化に Transformer を用いることの有効性や課題について考察する．

2. 非構造化データ

2.1 非構造化データ

非構造化データとは，特定の型やフォーマットが定められていないデータである．例えば文書，メー

ル，画像，音声などが非構造データに挙げられる．一方で，構造化データは CSV やデータベースなど，フォーマットが定められたデータである．構造化データは，行列で管理されているのですぐに活用できる状態にあるが，非構造化データを活用するためには，前処理を施す必要がある．

2.2 非構造化データの構造化における課題

非構造化データの構造化にあたって，下記のような課題が挙げられる．

2.2.1 必要情報の取捨選択

非構造化データにおける情報の重要度や関連性は，すべて均一である．助詞であっても固有名詞であっても，その重要度に優劣をつけるには，人間が文書をすべて読んで解釈するような高度な作業が必要となる．

2.2.2 文章表現の統一

非構造化データは構造化データとは異なり，あらかじめ定められた型や表記などがないため，その表現方法は多種多様である．同一の意味を表す文章であっても，単語や文字一つの順序によって表現は大きく異なるため，表現の統一が必要となる．

3. 構造化タスク

本研究では，次に示すようなメール文書（表 1）を JSON（表 2）に変換することを考える．メール文書には求人情報が書かれており，案件名や内容，勤務地などの文章が記載されている．対して JSON（表 2）では，「”job_name”:[AI Web アプリ開発案件]」などのように Key : Value の形で，メール文書の内容が書かれている．Key がメール文書から抽出したい項目名，Value がその内容となっている．次に具体例を交え，本研究で取り組んだタスクを紹介する．

3.1 特定個所の抽出

抽出すべき項目名とその内容の関連性を判断して，特定の個所を抽出する．例えば，メール文書（表 1）から「job_name」（表 2）に対応する内容である「AI Web アプリ開発案件」という文を探し出し，job_name の Value に格納する．

3.2 表記変換

同一の意味を持つ様々な表記を統一する．メール文書内（表 1）の「【単価】」に記載されている「50～60 万前後」を JSON（表 2）の「"price_min": [500000], "price_max": [600000]」にあるような数値に変換する．また，「【推奨スキル】javascript, django Web アプリ開発の経験」内の「javascript」（表 1）などの単語を「"skill_recommended_web": ["JavaScript", "Django"]」（表 2）のようなパスカルケースに変換することなどもこれに含まれる．

表 1 メール文書の例

【案件名】 AI Web アプリ開発案件
【内容】 チャットボットを用いた Web アプリの設計・開発に携わっていただきます
【勤務地】 茅場町 （東京メトロ東西線・日比谷線）
【期間】 10 月～12 月 ※延長の可能性あり
【必須スキル】 Python, C#を使用した開発経験 自然言語処理に関する知識
【推奨スキル】 javascript, django Web アプリ開発の経験
【単価】 50～60 万前後
【年齢】 40 歳はじめまでを希望
【人数】 1 名
【面談回数】 2 回（弊社同席）
【その他】 勤怠，コミュニケーションに問題のない方

表 2 JSON の例

{"job_name": ["AI Web アプリ開発案件"],
"contents": ["チャットボットを用いた Web アプリの設計・開発に携わっていただきます"],
"sites_station": ["茅場町"],
"durings": ["10 月～12 月"],
"price_min": [500000],
"price_max": [600000],
"age_min": [-1],
"age_max": [43],
"skill_required_os": [],
"skill_required_lang": ["Python", "C#"],
…,
"skill_recommended_web": ["JavaScript", "Django"],
"skill_recommended_tech": [],
"required_numbers": ["1 名"],
"counts_for_interview": ["2 回（弊社同席）"],
"etc": ["勤怠，コミュニケーションに問題のない方"]}

3.3 表現の統一

非構造化データ，特に人間が作成したデータには曖昧な表現がみられる．例えば，メール文書内（表1）の「【年齢】40 歳はじめまでを希望」は，何歳までのことを指しているのだろうか．41 歳までだろうか，43 歳までだろうか，それとも 45 歳までだろうか，非常に判断が難しい．こうした抽象的な表現を定量的な表現に統一する．ただし，定量的な表現の統一方法は，学習データに依存する．

4. Transformer

メール文書から JSON への変換にあたって，機械翻訳などに用いられる Transformer を採用した．次に Transformer の特徴を簡単に紹介する．

4.1 アーキテクチャ

Transformer は Encoder と Decoder，二つの要素から構成される[2]．翻訳元の文をソース文，翻訳後の文をターゲット文とすると，Encoder はソース文における単語間の関連性に関する情報をエンコードする．Decoder は，ターゲット文における単語間の関連性に関する情報を作成し，それと Encoder で得た情報を掛け合わせ，翻訳文を予測する．Encoder は Self Attention を，Decoder は Self Attention と Encoder-Decoder Attention を持ち，この Attention が文脈の依存関係を学習する．

4.2 Attention メカニズム

単語間の関連性を表す Attention は次の計算式によって算出される[2]．Q,K,V は文章を分散表現で表したしたベクトルである．

$$Attention = softmax(\frac{QK^T}{\sqrt{d_k}})V$$

Q と K の内積は，文内の各単語の類似度を意味する．d_kは中間層の次元数である．d_kが大きな値である場合，Q と K の内積が大きくなりすぎるため，softmax 関数の勾配は極端に小さくなってしまうという問題がある．そのため，$\sqrt{d_k}$で内積を割ることによって調整する．こうして算出された値は，softmax 関数で単語間の関連性を表す確率に変換される．

4.3 Multi-Head Attention

Transformer は Multi-Head Attention と呼ばれる複数の Attention を並列に用いることによって，高精度の翻訳を実現した．これは各々の Attention が異なる位置の異なる部分に注意を向け，単語間の関連性を処理するためである．Multi-Head Attention は次の式で表せられる[2].

$$MultiHead(Q,K,V) = Concat(head_1,\dots,head_h)W^O$$
$$where\ head_i = Attention(QW_i^Q, KW_i^K, VW_i^V)$$

$$Where\ the\ projections\ are\ parameter\ matrices$$
$$W_i^Q \in \mathbb{R}^{dmodel \times d_k}, W_i^K \in \mathbb{R}^{dmodel \times d_k}, W_i^V \in \mathbb{R}^{dmodel \times d_v}$$

ただし，h は Attention のヘッド数，WQ,WK,WV は重み行列，d_{model}は中間層の次元数，d_kはd_{model}を h で割った Q と K の次元数，同様にd_vは V の次元数を表す.

4.4 Positional Encoding

Transformer は入力シーケンスの単語順序を扱うために，Positional Encoding を用いてシーケンス内における単語の位置情報を付与する[2]．下記に示す正弦関数と余弦関数が入力シーケンスの分散表現と同じ次元数の分散表現を作成し，入力埋め込みに加算することによって，位置情報が分散表現に付与される．

$$PE_{(pos,2i)} = \sin(pos/10000^{2i/dmodel})$$
$$PE_{(pos,2i+1)} = \cos(pos/10000^{2i/dmodel})$$

ただし，pos は単語位置，i は次元数を表す.

4.5 Transformer を採用した理由

非構造化データを構造化データとする上で，課題となるのは，(1)不必要な情報と必要な情報を判断し抽出すること，(2)同一の意味を持つ多種多様な表現を矛盾なく統一することである．Transformer には Attention メカニズムがあり，Self Attention は文章内の単語間の関連性から重要な情報とそうでない情報を判断することができる．また，Encoder-Decoder Attention は，ソース文の単語とターゲット文の単語間の関連性を求め，多種多様な表現を統一するのに有効と考えられる．加えて，Transformer は End-to-End

学習であり，ルールベースでは対応しきれない様々なケースに対応できることが期待される．

5. 実験

5.1 使用データ

本研究で使用したデータは，求人情報に関するメール文書と，メール文書を元にルールベースで作成した JSON からなる．データ数は約 48,000 件，内 80%を学習用，10%を検証用，10%をテスト用のデータとした．メール文書，JSON はどちらもテキストに変換し，MeCab を用いて単語に分割した．JSON における単語分割に際しては，項目名（Key）はそれ自体が最小単位である一単語として扱った．各文のシーケンスの長さは，対象データ内における最長の 1000 に固定した．下記にメール文書と JSON の各シーケンス長の統計量を示す．

表 3 メール文書 シーケンス長の統計量

データセット	平均	中央値	標準偏差
学習	231.6	198.0	128.5
検証	231.0	199.0	127.8
テスト	229.0	197.0	125.3

表 4 JSON シーケンス長の統計量

データセット	平均	中央値	標準偏差
学習	253.4	238.0	70.4
検証	254.0	238.0	70.8
テスト	252.7	237.0	69.8

学習データ，検証データ，テストデータの統計量に大きな違いは見られない．また，メール文書，JSON における各語彙数は，35,201 個，24,165 個である．

5.2 パラメータ

モデルの学習にあたって使用したパラメータを下記に示す．

表 5 パラメーター覧

パラメータ	数値
バッチサイズ	64
レイヤー数	4
ヘッド数	4
中間層の次元数	64
Feed Forward Neural Network の次元数	2048

6. 結果

精度評価には，約 4,800 件のテストデータを使用し，予測によって得られた JSON と正解コーパスの JSON において，同じ項目の値を Exact Match と F 値で評価したところ，それぞれ、90.44 と 96.59 という結果となった．

次に，テストデータの推論によって得られた Attention Weight の出力結果から，非構造化データの構造化における Transformer の考察を行う．なお，以降に示す Attention Weight の図は，説明のために一部を切り出して掲載している．

6.1 特定個所の抽出

下記に示す図 1 は，job_name における Attention Weight の例である．横軸が入力文，縦軸は推論結果であり，入力文の各要素と推論結果の各要素の関連性を表している．色が明るいほど，関連性が高いことを表す．

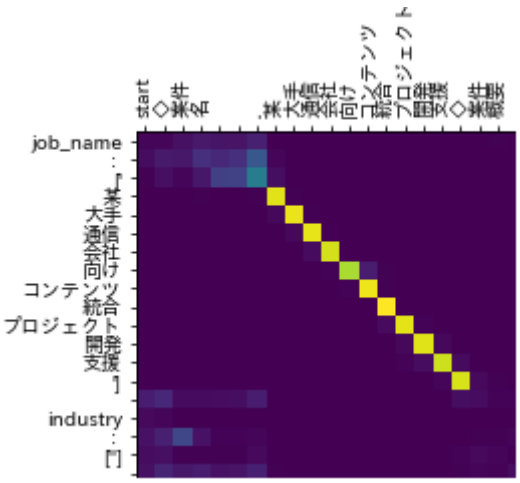

図 1 job_name における Attention Weight

これを見ると「コンテンツ」や「プロジェクト」など JSON の Value にあたる要素の関連性が高く，一方で，Key である「job_name」と Key にあたる「案件」や「名」の関連性が低い結果となっている．つまり，Value における関連性は判断できているが，Key と Key にあたるものの関連性は判断できていない．これは，学習時のメール文書と JSON に同じ要素が含まれているため，同一の要素に対する関連性を強く学習している結果と考えられる．したがって，特定個所の抽出において，どこからどこまでを抽出すべきかの判断材料となっているのは，Value 要素であると考えられる．

6.2 表記変換

下記に「price」の例（図 2）と「skill_required_lang」（図 3）の例を示す．図 2 を見ると，特定個所の抽出と同様に，関係がありそうな「単価」や「price_min」，「price_max」の関連性は出ておらず，数値（金額）における関連性が高く出ていることがわかる．

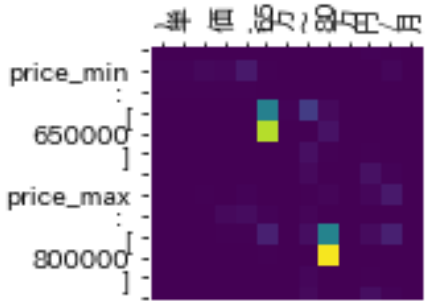

図 2 price における Attention Weight

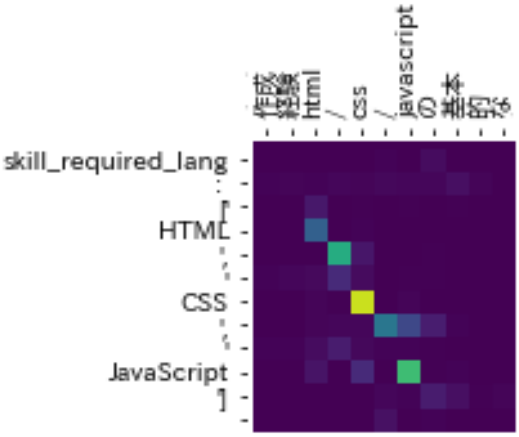

図 3 skill_required_lang における Attention Weight

図 3 の「skill_required_lang」も同様に，「HTML CSS JavaScript」など JSON の Value 要素の関連性が高く出ている．また，入力文における「/」と JSON の Value 要素の「,」の関連性も見られ，区切り文字として対応付けがなされていると考えられる．

6.3 表現の統一

次に「age」の例を示す．図 4 を見るとやはり JSON の Value 要素に強い関連性を示していることがわかるが，図 5, 6 では，「くらい」，「まで」，「〜」などの要素が下限値がないことを表す「[-1]」の「[-」部分に対して関連性を示しており，図 4 の「年齢 20 歳〜27 歳…」ように下限値が明示されていない場合は，そういった表現を吸収していると考えられる．

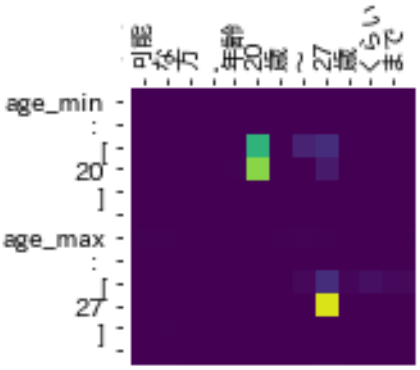

図 4 age における Attention Weight

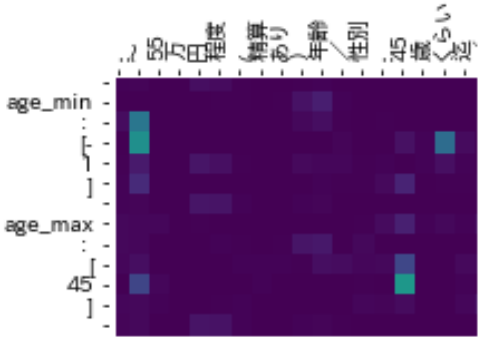

図 5 age における Attention Weight

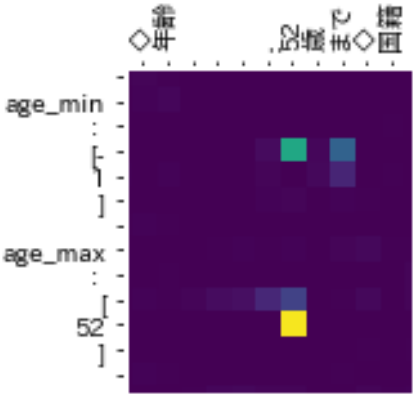

図 6 age における Attention Weight

6.4 課題

本稿で用いた Transformer では，メール文書と JSON の Value との関連性に着目して予測を行っているため，表記変換は得意であるが，表現の統一は不得意である．それは，文章における表現というものが複数の単語から構成され，その並びによって文脈が形成されるからである．例えば，下記のケースはどうであろうか．

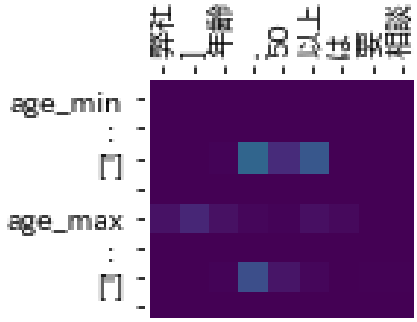

図 7　age における Attention Weight

入力文の「年齢.５０以上」まで見ると，「age_min:[50]」と変換したくなるが，その後の「は要相談」まで含めると，「age_max:[50]」が正しい推論結果となることがわかる．このように，表現の統一を行うためには，1 対 1 の単語間における関係性だけではなく，複数の単語間の関係性から文脈をとらえる能力が必要となる．

また，本稿におけるもう一つの課題は，非構造データの構造化を翻訳タスクとして扱っている点である．翻訳タスクとして扱う以上，入力単語と出力単語が同一であっても，内部では日本語から英語に翻訳するような処理が行われている．そのため，特定個所の抽出であっても意図しない単語に変換される場合がある．

7.　おわりに

本稿では，Transformer による非構造化データの構造化手法について紹介した．非構造化データにおける特定個所の抽出や表記変換などは，Transformer を用いることの有効性が確認された．しかしながら，非構造化データにおける表現の統一はいまだ課題があり，より高度なコンテキストを扱うことができる BERT を用いた研究の必要性を感じる．また，本稿では，非構造化データの構造化を翻訳タスクとして扱ったが，翻訳タスク故の副作用も見られたため，機械読解による情報抽出など別のアプローチも視野に入れつつ，課題に取り組んでいきたい．

参考文献

[1] 総務省，平成 29 年版情報通信白書，第 1 部 2 章 2 節 データ流通利活用における課題, pp.74, (2017)

[2] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, Illia Polosukhin,: Attention Is All You Need