平成 30 年度

筑波大学情報学群情報科学類

卒業研究論文

題目
モジュール分割された日本語対話システムの研究

主専攻　　知能情報メディア主専攻

著者　　江畑 拓哉

指導教員　Claus Aranha 櫻井鉄也

要　旨

自然言語処理の一分野に対話システムという分野がある。昨今このシステムは複数の小問題を解決するシステムから構成されることが多く、それぞれのシステムにルールベース・機械学習問わず様々な手法が用いられている。本研究では特に日本語を対象とした対話システムを構築することを目指して、大まかな全体像を描いた上で、それに必要となる手法について、主に深層学習の手法を用いてそれぞれ研究を行う。またこのようなシステムを構築する際に不可欠になるデータの収集法と、その前処理について調査する。
具体的には発話データ・対話データの収集法とそれぞれの収集したデータの性質調査、短文データの偏りがある場合の分類手法の研究、深層学習を用いた対話モデルの手法の比較、文のスタイルを変換する手法の比較、深層学習を用いた短文生成モデルに対するエラー検知についての研究を行った。

Contents

List of Figures

1 序論

1.1 研究背景及び目的

ある目的に対してより完璧に (accuracy が高くなるように) 命令を実行をする Artificial Intelligence が求められている昨今の AI 競争の時代に対し、自然言語処理やゲーム AI のようなタスクは極めて複雑な課題を抱えている。例えばそれは “言葉” という問題である。これは人間がコンピュータに正解となるものを提供することが極めて難しく、“なんとなく良い感じに” 目的を達成してくれることを期待することが多い。この問題に対処するための手段として、入手でき得る限りの大規模なデータを用意して中心極限定理的に尤もらしい中心部を得る方法や、とにかく何らかの単一のモデルに押し込めて問題を解くという方法 [1] がある。それに対して、データそのものを一旦精査・前処理すること、問題を整理・分解しそれぞれを解くことも研究として存在している。

尚、日本で人気を得ている “マルチモーダルエージェント AI” は複数のソースから問題を見直すという特徴があるが、これは複数のモデルを使っているという意味で同じではあるが、問題を分割しようとしているわけではないという点でこの研究と大きく異なる。

自然言語処理の、特に対話システムについて考えたとき、小問題に分割した上で対話システムを達成した例として、例として Amazon Alexa Prize [2] というコンテストや Microsoft 社が研究・開発している “りんな” [3] を挙げることができる。これらは対話を行うという問題に対して小さな部分問題を解くタスクを設定し、それぞれを組み合わせることで元の問題を解くというスタイルを取っている。

本研究ではこれらを参考に、日本語の対話システムを作成するという問題に対して小問題を設定しそれを解くための手法を提案・実験する。またその前準備としてデータ収集に絡めて日本語データとその前処理について考察する。尚本研究が最終的に望むものは、キャラクタ性を持った対話可能なエージェントを作ることであることを強調する。

もう少し言及すると、本研究ではデータからモデルにかけて 5 つの少テーマについて研究を行った。概要をそれぞれ説明すると以下のようになる。

1. 日本語データの取り扱いについて
   我々は一般に日本語を話しており、それを用いた対話システムの構築が本研究の主目的である。しかし機械学習等のデータセットや実験で多く使われているのは、日本語とは使っている文字や文型で大きく異なっている、英語のものが多い。その前提のもとで日本語のデータ、特にセンテンスに対して、どのような性質があるのかを調査し、また提案する漢字→ひらがな変換という前処理とそれによって得られる性質についても議論を行う。

2. 文抽出を念頭においた不均衡分散・サイズの分類問題
   テキストのカテゴリ分類を考えたとき、一般にはおおよそ同程度のサンプル数が期待できる n 個のカテゴリの中から任意の文が入力されることを想定している。今回はそれとはやや問題設定が異なり、いくつかの文をカテゴリ $1 \cdots n-1$ 、それ以外のすべてををカテゴリ n として扱うことについて考える。そうすると、カテゴリ n のみ異様に得られるサンプル数、分散が大きくなってしまう。本テーマではこの問題について議論を行う。しかしデータを設定・収集することが困難であったため、一部画像認識の問題に置き換え実験を行った。

3. 機械翻訳システムを用いた対話モデル
   一対一対話を行う際に、機械翻訳システムを用いることがある。今回はそれを、問題を文脈に依存しない発話に対する反応を学習することに再設定し、Transformer という 2017 年 12 月時点で SOTA (State-Of-The-Art 最高水準) を獲得した機械翻訳の手法を用いて実験、有名な手法である Sequence to Sequence Attention を用いた一対一対話モデルと比較を行う。

4. 文のスタイル変換
   文のスタイルとは、例えば口調や訛り、書き言葉や話し言葉といったものを指す。これは日本語で特に顕著に見られるもので、テキスト上でもこれを確認することで相手のペルソナをある程度想定することができる。本研究が日本語を対象としていること、キャラクタ性を持たせたいというモチベーションがあることから文に特定のスタイルを持たせることを問題として取り上げる。

5. CoLA タスクを応用した対話システムのエラー検知
   対話システムを小問題に分割して解く弊害として、それぞれの問題でエラー (不適切な出力) が出て

[1] HRED (Sordoni et al. 2015) や VHRED (Serban et al. 2016) があるが、発話の多様性を得ること (一般的な受け答えを学んでしまい、同じような文ばかり生成してしまう) やデータを十分に集めることが難しいなど課題がある。

[2] https://developer.amazon.com/alexaprize

[3] https://twitter.com/ms_rinna

しまうというものがある。これに対処するため、特に何らかの機械学習モデルから生成された文に対しそれが自然であるかどうかを評価するモデルを作成し実験する。

1.2 本論文の構成

第 1 章に本論文の概要とその構成について説明を行い、第 2 章で関連研究を紹介し、第 3 章で本研究で掲げるシステムの全体像を示す。そして第 4 章から第 9 章にかけては 1.1 で述べたテーマについての順に議論する。その後付録として補足をまとめたものを 第 10 章 として示す。最後に議論として本論文のまとめ、今後の展望について述べる。

2　対話システムの関連研究

対話システムの関連研究としては、1.1 で述べたように Amazon Alexa Prize というコンテストや、Microsoft 社のりんなを挙げることができる。

Microsoft 社のりんなは日本語雑談対話 (Wo et al. 2016) を実現しており、2018 年現在 Twitter [4] などで活動をしている。

Amazon Alexa Prize は Amazon Alexa という音声会話を行うことのできる端末に搭載する対話システムを競う大会である。評価対象はユーザの印象であり、別の指標として対話時間が公開される。2017,2018 年度の Amazon Alexa Prize では平均 10 分程度の対話を行うことの出来たシステム、Sounding Board(2017 年度)、Gunrock(2018 年度) が優勝した。顔や体といったテキスト以外の情報を用いることの出来ない対話システムでこのような結果が得られたことは注目すべきことである。(しかしこれらは極めて大規模なデータを元にして作成されているため、これを個人で実装することはほぼ不可能だ。)

いずれも複数のモデルを組み合わせて構成されており、例えば言語理解部と文生成部、そして本研究で取り扱わないものとしては、音声理解部と音声生成部を挙げることができる。またりんなに関してはそれに加えて画像認識部などの対話以外の [5] システムも構築している [Figure 1] 。

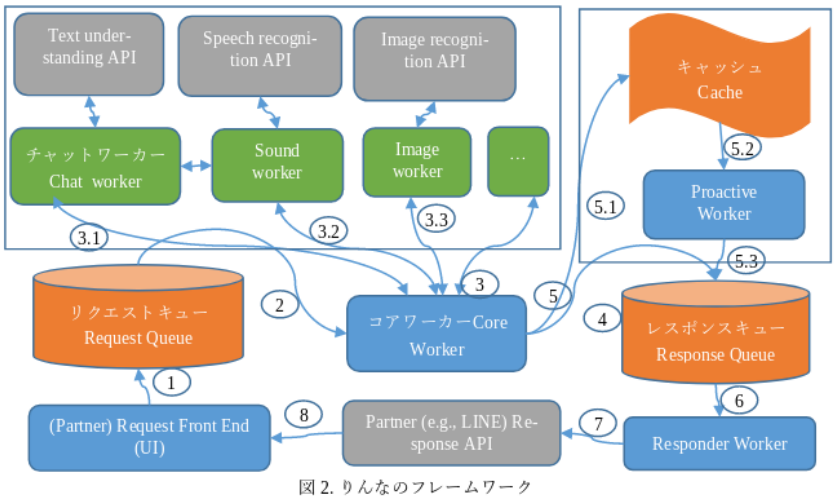

Figure 1:　りんなのフレームワーク (Wo et al. 2016 より)

[4] https://twitter.com

[5] 対話をテキストやそれを示す音声のみのコミュニケーションと定義した場合。実際には対話には身振り手振り、表情といった要素が複雑に絡んでいる。そのため 2017 年頃からは、表情を考慮した対話システムが提案され (Chu, Li, and Fidler 2018) 研究されている。

2.1 Sounding Board

Sounding Board (Fang et al. 2018) は 2017 年度の Amazon Alexa Prize で優勝した social bot の名称で、Washington 大学の大学院生らが作り上げたシステムだ。ここで定義する Social bot とは Personal Assistant と所謂 ChatBot (日本で言う ChitChat 或いはチャットボット、人工無能) の中間にあるものだ。本研究と異なるものとして、これが目的としているものは “bot” の開発で、“人と対話するよな” 体験をさせることを目的とするわけではなく、例えば受付や何らかの教師といったシステムへの応用を考えていることが挙げられる。

また Sounding Board はネットワークを介して積極的に外部情報を用いるという点が特徴だ。また逐次的に収集される情報を知識グラフとして持つことで、過去の資源とのつながりも持つことが出来ると考えられる点も興味深い。これらの機能のおかげでシステムの老朽化を比較的に抑えることを見込まれている。

更にユーザモデリングを重視している点や、ユーザの発話から会話を掘り下げていくスタイルはこの bot が審査員から 10 分以上の対話を勝ち取った秘訣として挙げることが出来る。しかし逆に言えばこの bot はキャラクタ性や自主性に乏しい。その点が本研究とは目標が異なっているものとして挙げられる。

また小規模のルールベースを用いたモデルを用いているという点は非常に興味深い。本研究では深層学習を積極的に用いるようにしているが、こちらはデータの前処理が苦になるということを Sounding Board の研究では指摘している。この点については先述の章らで示したように認めざるを得ない。

[Figure 3] に Sounding Board のシステムアーキテクチャを引用する。まず Front-end であるが、これは Amazon が提供している Automatic Speech Recognition (ASR) [6] と Text-to-Speech (TTS) API [7] が用いられている。Middle-end に関しては記載されている通り、NLU(Natual Language Understanding) と Dialogue Management、Natural Language Generation の 3 テーマをそれぞれ小問題に分割して解決している。Natural Language Understanding ではユーザからの入力の振り分け、感情分類などを行っており、Dialogue Management では会話全体を管理するマスターと、挨拶や特定のトピックに対する対話など様々なタイプのモードについての処理を行う集合とによる階層構造を持っており、マスターでは会話の一貫性やユーザからのエンゲージメント、コンテンツの可用性などを推測し会話の統制を取っている。Back-end では主に知識グラフを保存するために用いられている。

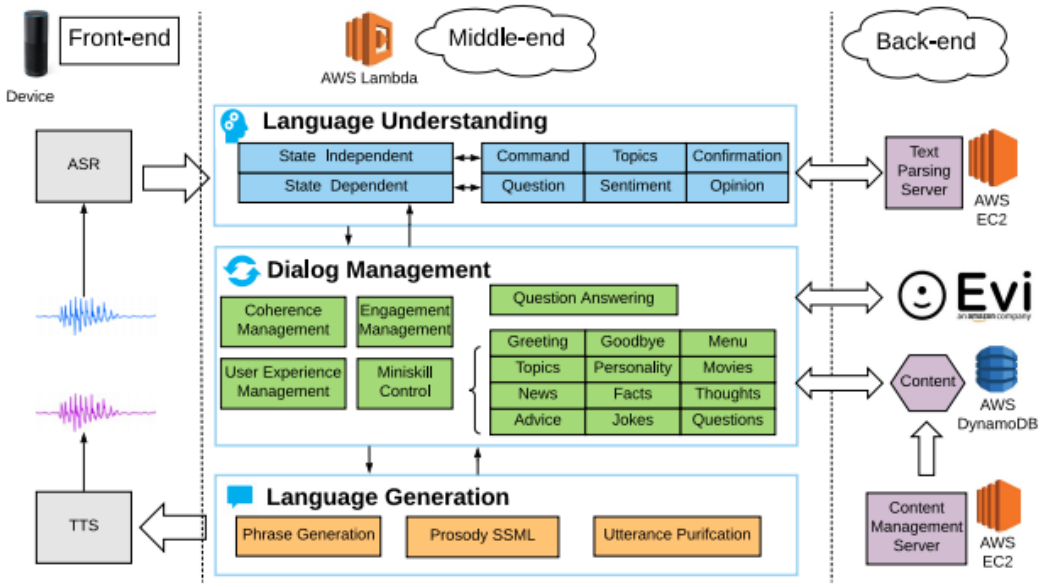

Figure 2: Sounding Board のシステムアーキテクチャ (Fang et al. 2018 より)

[6] https://developer.amazon.com/ja/alexa-skills-kit/asr

[7] https://aws.amazon.com/jp/polly/ や https://developer.amazon.com/ja/docs/alexa-voice-service/speechsynthesizer.html

2.2 Gunrock

Gunrock (C.-Y. Chen et al. 2018) は 2018 年度の Amazon Alexa Prize で優勝した Social bot の名称で、Calfornia 大学の Davis 校のチームが作成した Social bot である。

この研究では特に動物、映画・本、音楽といったトピックごとの対話を独立の流れを持つものとして取り扱ったこと、その bot の性格や好みを処理できるようにしたこと、NER について議論していること、NLP における相互参照について処理したことが注目できる。1 つ目に対してはトピックを分割するべきという立場が本研究に近いものとして考えることが出来る。2 つ目に対してはは本研究の求めるものに非常に近い。しかしこれに関しては特定の質問を抽出する際に文の類似度を計るという解決策を取っていたものの、その手法にやや問題のあるもの [8] を用いていたため本研究では参考とすることが出来なかった。3 つ目に対しては特に英語の大文字小文字が NER に与える影響について言及しており、この問題は本研究で日本語の漢字→かな変換との関連を意識することが出来る (実際にはこの論文が発表される前に提案していたため、直接の関連はない)。4 つ目に対しては特に日本語の NLP で極めて重要になる課題であると考えることが出来るものの、本研究では議論することが出来なかった。

また本研究とは接点がないものとしては音声理解の分野や、発話音声の抑揚などの調節について提案しそれが有効であることを示したことを挙げられる。本研究はキャラクタ性を重視していることから、発話の実装を検討をした場合、発話音声の調節は極めて重要な話題であると考えられる。

[Figure 4] に Gunrock のシステムアーキテクチャを引用する。ASR や TTS は Sounding Board と同じものを指し示している。Netural Language Understanding は 3 層の構造体になっている。1. Segmentation は、句読点を挿入して文のような単位に分割するということを行う部分、 2. Noun Phrase は、相互参照を解決する足がかりとなる、代名詞句や名詞句を取り出す部分、 3. NER や Coreference などは、対話のログや知識ベースを用いて取り出した代名詞句・名詞句の補完を行う部分だ。Diglogue Manager は、対話の流れを汲み取り Topic Dialogue Module へと繋げる処理と担う Intent Classifier と、それに応じて対話内容を作成・Intent Classifier にフィードバックする Topic Dialogue Module の 2 つで構成されている。Natural Language Generation は生成された応答内容のチェックや文章化するためのテンプレートの提供、音声調律などを行う。Backstory はその bot そのもの、 bot の個人的な話題を処理するためのもので、EVI は所謂常識についての質問を処理するもので、例えば史実といった質問に対する回答を提供する。Knowledge Base は Reddit や Twitter などのオンライン資源からのデータをトピックごとに保持している。またそれらのデータは知識グラフに統合されてるようになっており、Sounding Board で紹介したような利点を見込むことが出来る。

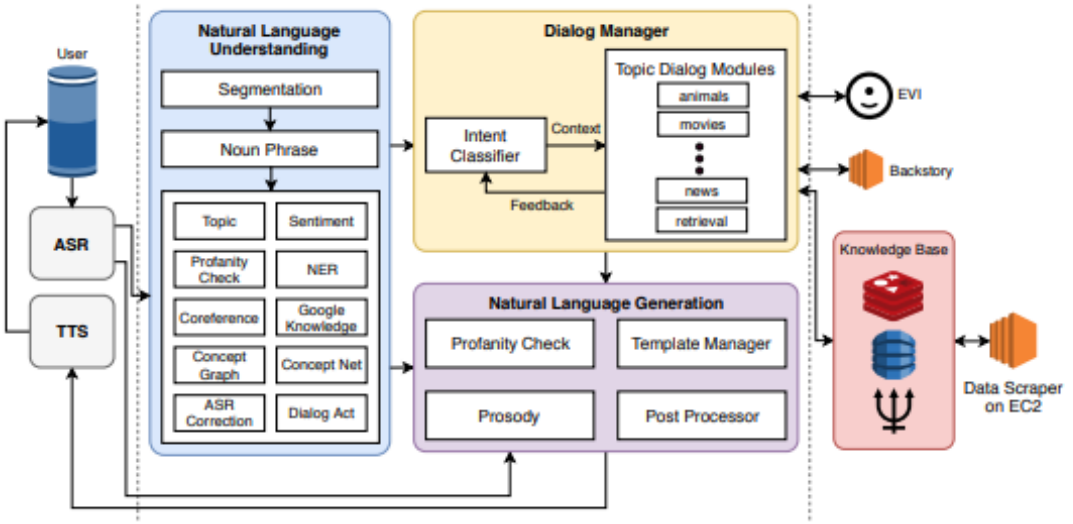

Figure 3: Gunrock のシステムアーキテクチャ (C.-Y. Chen et al. 2018 より)

[8] Universal Sentence Encoder と呼ばれるモデルで、Google 社の研究成果であるものの、性能や論文の内容について大変評価が悪いことで有名だ。https://www.reddit.com/r/MachineLearning/comments/88c2vp/r_180311175_universal_sentence_encoder/

3 想定する対話システムの全体像

以下に本研究で想定する対話システムの全体像を示す [Figure 2]。

このシステムでは入力としてテキストと、環境情報を得る。このシステムにおける環境情報とはこのシステムが組み込まれているエージェントが居る場所の環境 (天候や気温・湿度)、エージェントの内部状態 (メモリ使用率等) を指す。これはテキストを用いた人対人の対話をイメージしたもので、つまり相手の居る環境、相手の体調をそれぞれ置き換えたものになる。また Answer Generation に用いる所謂個人データのようなものもエージェントの内部に持っているものとする。本論文で扱うものは、この内の Sentence Detection / Sentence Categorization / Topic Dialogue / Style Transfer である。また Topic Dialogue から Style Transfer への矢印・Answer Generation から Style Transfer への矢印・Style Transfer から Output への矢印におけるエラー検知についても議論する。

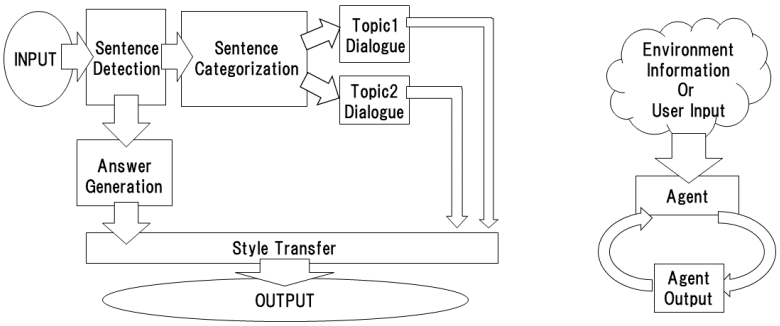

Figure 4: 本研究のシステム全体像

- Sentence Detection [該当部:文抽出を念頭においた不均衡分散・サイズの分類問題]
  ある特定の文を取り出す。取り出された場合はどの意味として取り出されたのかという情報とともに、Answer Generation へ向かい、取り出されなかった場合には付加情報なしで Sentence Categorization へ入力を受け流す。 最終的にはほとんどの文をここで抽出し、それに対する返答を Answer Generation でエージェントの内部状態ないし外部知識ベースを参照しながら生成する。

- Sentence Categorization [該当部:日本語データの取り扱いについて・文抽出を念頭においた不均衡分散・サイズの分類問題]
  文を大雑把にカテゴリ分類する。例えばそれは livedoor news corpus [9] で議論されるような スポーツ/IT/家電 といったようなカテゴリである。ここでカテゴリ分類された文はそれぞれ対応する Topic Dialogue に流される。

- Topic Dialogue [該当部:機械翻訳システムを用いた対話モデル]
  与えられたカテゴリに対する一対一応答を行う。例えばゲームについての話題を受け持つ Topic Dialogue はゲームに関する入力文を期待しており、それに対する出力を学習しているものとする。そのモデルはエージェントのペルソナに応じて置換することが可能であり、例えば好きなゲームカテゴリについての好意的なデータを多分に含んだデータセットで訓練した Topic Dialogue はそのゲームカテゴリが好きな (好きになった) エージェントが持つことになる。

- Style Transfer [該当部:文のスタイル変換]
  文のスタイルを変換する。ここで言う文のスタイルとは例えば書き言葉や話し言葉、各ペルソナに基づいた語尾変化を示す。

- エラー検知についての議論 [該当部:CoLA タスクを応用した対話システムのエラー検知]
  上記のシステムで発生するエラーデータと正常なデータを分類する。

[9] https://www.roundhuit.com/download.html#ldcc

4 日本語データの取り扱いについて

日本語データは英語データに比べていくつかの問題を抱えている。問題の例としては、文字の数が多すぎること、スペースといった意味ごとの分割がないこと、容易にペルソナを特定できるような多彩な語尾変化があること、多国語も日本語であるかのように用いること、同意同音の語でも様々な表記方法があることが挙げられる[10]。

また一般に公開されている対話データセットを対話テキストのみで学習させると想定したとき、背景知識の欠如を指摘せざるを得ない。更に言えば日本人の特徴として “言外にわかり合う” というコミュニケーションスタイルも問題を難しくしていると言えるだろう。

この章では上記の問題があることを公開されているデータセットや Twitter から収集したデータセットを用いて調査するとともに、“漢字をかなに変換する” という前処理を用いることでどのようにデータの性質が変化するのかを、単語分散を得るというタスクについて実験する。

尚本研究では、形態素解析には MeCab[11] 0.996、単語辞書として mecab-ipadic-neologd[12] 20181112-01 を用いた。特に Twitter のようなデータは流行語や新語に対応するため、単語辞書を定期的に更新する必要がある。

4.1 調査 1) 発話データの性質

発話データとして、2018 年 12 月 25 日 23:00 頃 から翌 26 日 10:00 頃 までに収集した 7 万件の Twitter データを収集し、その性質を観測した。

データの収集手法としては Twitter 社が公開している API を用い、日本のユーザから呟かれている内容を集めるものとした。この処理によって生データが 77,285 発話得られた。

4.1.1 フィルタ

データを収集するにあたり、タグや宛名、URL リンクと言った Twitter に特有な部分を省いた。その上で、4 文字以上、60 文字以下のデータをすべて抽出し、データを 54,368 発話にした。

Twitter に特有な部分を省いた理由として、全体の目的から考えて Twitter データに特化させる必要がなかったこと、タグは時系列で発生・消滅すること、宛名に関してはそのユーザの背景情報が必要になることが容易に想像できること、URL リンクを発話として認めるべきではないと考えたことを挙げる。

勿論いくつかの懸念事項は存在する。例えばタグに意味が込められている例 (“#○○を許すな” など) が少なからず見られたが、タグを認めるとタグのあるすべてのデータを手動で確認する必要があったため今回はすべて省いた。

また文字数でフィルタを行った理由として、1. 4 文字未満のデータは少なく、この後議論する単語分割が出来ないようなデータ、それのみでは意味が通じないデータが多く含まれていたこと、2. 60 字超過のデータは何らかの内容に対する説明と言った発話データとはややベクトルの異なるデータが多かったこと、深層学習を中心とした機械学習を用いた自然言語処理 (要約タスクを除く) に用いるデータであると考えたとき、長すぎるテキストはその一部を短くする前処理が施されることが一般的であること、を挙げる。

Table 1: 発話データに対して適用したフィルタとその理由

フィルタの概要	詳細	理由
Twitter 特有の内容	タグ	時系列で発生・消滅するため
	宛名	宛名のユーザに対する情報が必要であるため
	URL リンク	リンクを発話として認めるべきか議論の余地があるため
文字数	4 文字未満	データ数が少なかったため
		単語分割が出来ないため (極端な略語など)
	60 文字超過	発話データというよりは説明のようなデータが多かったため
		適用する予定の手法では情報の一部が切り落とされてしまうため

[10]前 2 つに関しては、中国語も共通して抱えている問題と言える。

[11]http://taku910.github.io/mecab/

[12]https://github.com/neologd/mecab-ipadic-neologd

4.1.2 調査結果

フィルタによって抽出された 54,368 発話を調査した。
まず発話データとして問題があると考えられる発話について報告する。尚すべての報告における例は、個人情報を含んだ部分を含まないように編集されている。

Table 2: 発話データの調査結果 1

概要	詳細	例
他国語を用いた発話	中国語・英語等を用いた (含まれる) ツイートが 0.5 % 程度見られた	Very nice Merry Christmas! 謝謝 Guten Morgen!
テキストのみでは 理解できない発話	画像などのコンテンツに 対する発話が微量見られた ハイコンテクスト過ぎて 理解できないものが見られた	これ最高 れ!!!
(意図的・意図的でない) 誤字		オフトゥン イケメソ
顔文字や絵文字の多用	Twitter で許可されている絵文字や、 顔文字が含まれる発話が 8% 程見られた	(⋆´ω ｀⋆) お疲れ様です [(:3[■■]] (´∀ ｀)>
単語の一部や 語尾の繰り返し	特に感情的なつぶやきでは、 強調などの目的から 語の一部を繰り返す傾向が見られた	全全全休 ほにゃほにゃほにゃほにゃする やだぁあぁぁあぁぁあ!
略語の多用	長い単語、文は相互に理解できるような 形に省略されることが多かった	メリクリ! なるはや
別の表現	同じ意味を示すが 別の表記法があるものは 共通化されているわけではなかった	!/!!!!/！/ ‼ /!!!!!!!!! … /… こんど/今度 彼氏/カレ氏/カレシ デス/です
伏せ字	隠語など伏せ字を用いている場合があった	○ね
語尾の特徴付け等		ねれないぽよ ...と思うニョロ むいねー

次に主に情報の価値として問題があると考えられる発話について報告する。

Table 3: 発話データの調査結果 2

概要	詳細	例
個人情報の入ったもの	電話番号や SNS の ID などを 含まれるものが、 一万件に対して 5,6 件あった 個人名・アカウント名が含まれるものを 含めると 5%程になってしまった	
時刻など		2018.12.26 06:00
頻度が高すぎるもの	挨拶等	メリクリ！ おはよう
センシティブなもの		
Twitter 特有のもの		凍結された フォローありがとうございます

次ページに続く

前ページからの続き

概要	詳細	例
数値データ	英語での NLP の一部では積極的に削除されている 漢数字 ギリシャ数字	2018 200 円 一 V

最後にこの後実験として取り上げる極性判定のデータとして問題があると考えられる発話について報告する。

Table 4: 発話データの調査結果 3

概要	詳細	例
予定などのメモ書き	個人の予定や イベントの告知	
企業などの広告		
取引などのツイート		買) 鳥獣戯画のペンダント
豆知識や引用	特に深夜～早朝にかけては 自動ツイートのような形式の 豆知識や引用の頻度が高くなっていた 最大では 3 %程がこれに含まれていた	丁字染ちょうじぞめ オロバス ￥n ソロモン 72 柱の… [飲み会で使える？ダジャレ]… サーッ!(迫真)
感情が含まれているか 疑問のあるデータ		なぜ僕らは生きるのか

4.1.3 考察

データを収集した時間も相まって広告や豆知識・引用といった発話が多く観測された。これらのデータは極性判定やカテゴリ分類、ユーザクラスタリングなどに悪影響を与えることが論理的に考えられる。予定や広告、時刻などに関係したデータは、ほとんどの場合で一過性のものであるため長期的なシステムのためのデータとして見たときには適切であるか疑問が残る。

数値データや個人名のようなデータに関しては、英語での NLP、特に良い精度を持ったいくつかのタスクに対しては何らかの記号に置換されることが多い。しかし日本語でこれを適用しようとしたとき、1. 様々な表記方法があること、2. スペースで分割されていないため、形態素解析などの技術や NER(Named Entity Recognition 固有表現抽出) の技術を組み合わせなければ抽出できないこと、が問題として挙げられる。特に形態素解析に関しては Twitter のデータのような正規化されていないテキストに行った場合、精度が比較的に落ちるため、何らかの精度向上手法または別手法を提案する必要がある。

また同じ意味を表す文でも様々なバリエーションがあることがわかった。例えば “おはよう” を例に取ってみると、“おはようございます”、“おはよー”、“おは”、“おはよおおお”、“おは (愛称等)” といったバリエーションが見られた。これらはキャラクタ性を持たせるためには必要な分散であるが、意味のみに注目した場合や、語彙数の問題を考慮した場合には極力減らされたほうが良いと考えられる。これは英語の NLP (例えば機械翻訳) で前処理として、“he’s” を “he is” にするなどの前処理が行われることがあることからも推察される。更にバリエーションのある文は平均的に出現頻度が高いため、これを集めすぎるとデータに偏りが生まれてしまうことも考慮する必要があるだろう。具体的には、26 日午前 6 時ちょうど頃は 3 割程度が宛先や顔文字などの付加情報の差はあれど “おはよう” の意味の発話であったが、これをすべてデータとして認めてしまうと、この “おはよう” についてのデータが他のデータに対して極端に多くなってしまうことが考えられる。

極性判定のみに絞った議論をするならば、例えば自動ツイートされた発話にはユーザの極性があるとは考えにくいため、これを省くのが適当であると考えられる。しかし以上のことを踏まえてデータの再抽出・編集をフィルタリング後のデータの中の、15,000 程度のデータに対して行ったところ、1,500 程度のデータしか得られなかった。尚特にこの結果を招いた要因を挙げるとすれば、個人情報を含んだデータを編集・削除したこと、極性を持たないと思われるデータ (中性という意味ではない) を省いたことだ。

更に極性判定のためのデータとしてこのデータを考えると、顔文字や絵文字等は極めて感情を含んでいると感じられた。例えば、“おはようございます。(ノ∀ ` )” と “おはようございます。(* ´ ω ` *)” では極性判定上全く違う評価を下さざるを得ない。しかし顔文字や、特に絵文字については、そのバリエーションに際限がないことや機種依存文字などの入力可能性について議論しなければならない。これらを解消する

ためには、それらを例えば文字単位、或いはそれに準ずる単位で分割するなどしてある程度のカテゴライズを行えるようにする手法が要求される。

形態素解析で成功した例

りかちゃんありがとう

<形態素解析結果>
りか 名詞, 固有名詞, 人名, 名,*,*, りか, リカ, リカ
ちゃん 名詞, 接尾, 人名,*,*,*, ちゃん, チャン, チャン
ありがとう 感動詞,*,*,*,*,*, ありがとう, アリガトウ, アリガトー

形態素解析で失敗した例

山さんに …

<形態素解析結果>
山 名詞, 一般,*,*,*,*, 山, ヤマ, ヤマ
さん 名詞, 接尾, 人名,*,*,*, さん, サン, サン
に 助詞, 格助詞, 一般,*,*,*, に, ニ, ニ
…

※人名を指すが一般名詞として認識されてしまっている。
このよう場合には単語分割した後、NER を用いて検出することが望ましいと言える。

4.2 調査 2) 対話データの性質

対話データとして、2018 年 8 月から 12 月にかけて不定期に Twitter から収集した対話データ、一般公開されている書き起こしの対話コーパス、一般公開されているチャットの対話コーパスについてデータを観測した。

以下に調査結果として何らかの問題があると考えられる特徴について報告し、それに対する考察を述べる。

4.2.1 調査結果

1. Twitter から収集した対話データ

   収集方法は Twitter 社が公開している API を用い、日本のユーザから呟かれている内容の中から、3 発話以上対話が続いているものを収集した。この処理によって生データが 10,767 の対話ペアが得られた。そして生データに対しては 4.1.1 と同様にハッシュタグと宛名、そして URL リンクを削除したが、文字制限は対話間の意味を観測するため行わなかった。

Table 5: 対話データの調査結果 1

概要	詳細	例
センシティブな内容	3 %程はセンシティブな内容の対話であった。	
ゲームに関する内容	5 %程はゲームに関する内容であった。 その中には一過性の内容 (情報共有や待ち合わせ等) が含まれていた	
顔文字や絵文字等が含まれるもの	15 %程は顔文字や絵文字を含んでいた そのうちの 2 割ほどは顔文字・絵文字のみが 発話になっているものがあった	おはよーございます! ((⋆゜д゜)ノ ('д`)
似たような内容	特に挨拶など同じような 内容の対話頻度が高かった 朝方には半数が “おはようございます” の内容であった	おはようございますよ

次ページに続く

前ページからの続き

概要	詳細	例
事前知識を必要とする内容	間柄や話題 (例えばゲーム) の内容に 関する事前知識がいるものが 多く感じられた。[13]	line カメラたのしい
固有表現が含まれるもの	名前等固有表現が含まれるものは 3 割程度であった。	

2. 名大会話コーパスから収集したデータ

名大会話コーパス (逸子, 美恵子, and ディヴィッド義和 2011) から入手できる 129 会話について観測した。名大会話コーパスとは日本語母語話者同士の雑談を文字化したコーパスで、129 会話を収録、その合計時間は 100 時間に及ぶ比較的大規模なものだ。ライセンスがクリエイティブ・コモンズ表示-非営利-改変禁止 4.0 国際ライセンスで公開されているため、研究目的で用いることは非常に容易なコーパスであると言える。

非常に大規模かつ考察で述べるように複雑な内容であるため、出現頻度については言及しない。

Table 6: 対話データの調査結果 2

概要	詳細	例
言外のコミュニケーション	言語化せずに伝える内容があった	＜笑い＞ (共感の意)
長文や複文	相手が内容を理解したものとして 文を継続させる場合があった。	すごい勢いで走って。 私、あ、あーさっきの犬だとか 私たちが言っとるじゃん。 犬も気がついたじゃん。 じゃははって走ってきちゃって、犬が。
書き言葉・話し言葉の変化	あの → あん といった変化が見られた。	ほいでさあ、ずっと歩いていたんだけど、 そうすと上から、なんか町の中が見れるじゃん。
固有表現	個人情報保護のため 名前などの 固有表現は置換されていた	＊＊＊の町というのはちいちゃくって ... ほいで、あの、F023 さんはあたしが前の日に... Cが、あのー、写真を見せてくれたんだけど...

3. 対話破綻チャレンジの雑談対話コーパスから収集したデータ

対話破綻チャレンジ (東中, 船越, and 小林 2015) とは人間と対話システムとの間で生じる「対話破綻」(ユーザが対話を継続できなくなる状態) を自動検出することを目的とした、評価型ワークショップである。

このデータは対話システムと人間間とのテキストを用いた対話データと、その対話が成立しているかどうかを判定した複数人によるアノテーションが含まれており、本研究の目指すエージェントと人の対話の形に最も近いデータセットであると言える。

本データセットは問題点が少なく、アノテーションに従って、比較的成立しているとみなされた対話を抽出することで対話データを生成することが出来た。

4.2.2 考察

Twitter から収集した対話データに関しては Twitter データとして非常に有効であると考えられる。しかし比較的に個人的・センシティブな内容が多く、これを対話データとして学習させてしまうことによる、対話システムの倫理的な問題を考慮しなければならないだろう。また顔文字や絵文字等は 4.1 で考察したように単位で分割することが難しい。同様に同じような意味を持った対話が多く存在していたことから、これにも対処する必要があるだろう。

名大会話コーパスから収集したデータに関しては日常会話を分析・理解するには抽出するには非常に価値のあるデータセットであるが、これをチャットのようなテキスト入力等を介した対話には不適切なデータであると考えられる。このコーパスを観測して考察できる内容としては、1. 書き言葉・話し言葉の変化は想像以上に大きなものであったと言えること、2. 決して発話一つに対して返答が一つという形式になって

[13] アノテータが一人のため境界を判定することは難しいため、割合を明言することは出来ない。

いるわけではないこと、3. 固有表現の取扱についてより深く考察する必要があること、であった。
　対話破綻チャレンジから収集したデータはほぼ申し分ない自然さを持ったデータを集めることができることがわかった。しかし対話システムと人との対話データであるため、“人対人のような日常会話” 対話は比較的少なく、“人のような” 対話エージェントを作成するならば、不足している対話を外部から付け加える必要があると考えられる。

4.3　問題設定

　NLP の研究分野の一つについて単語分散を用いた言語モデル生成がある。単語といったある単位ごとの意味をベクトルなどの数値にする手法であり、この利点としては、単位ごとの距離を考えたとき、意味的に近い要素は近く、遠い要素は遠くなることで様々な NLP のタスクで自然言語を数値化する際に、自然言語の特徴を強く表すことができるようになるというものがある。例えば [Figure 5] では、4.5 から得られた単語分散を用いて犬と猫の名前ををいくつか描画した。これを見ると犬 (青) と猫 (赤) がうまく分離出来ていることがわかる。
　本テーマではこの単語分散を得るという問題に対してデータの前処理がどのように影響するのかを理解する目的で、2 つの実験を行う。
　一つは、1. 漢字・かな入り混じり文、2. かな飲みに変換した文、によって得られる単語分散の性質の違いを確認する実験、もう一つは得られた単語分散を用いて極性判定を行う実験である。

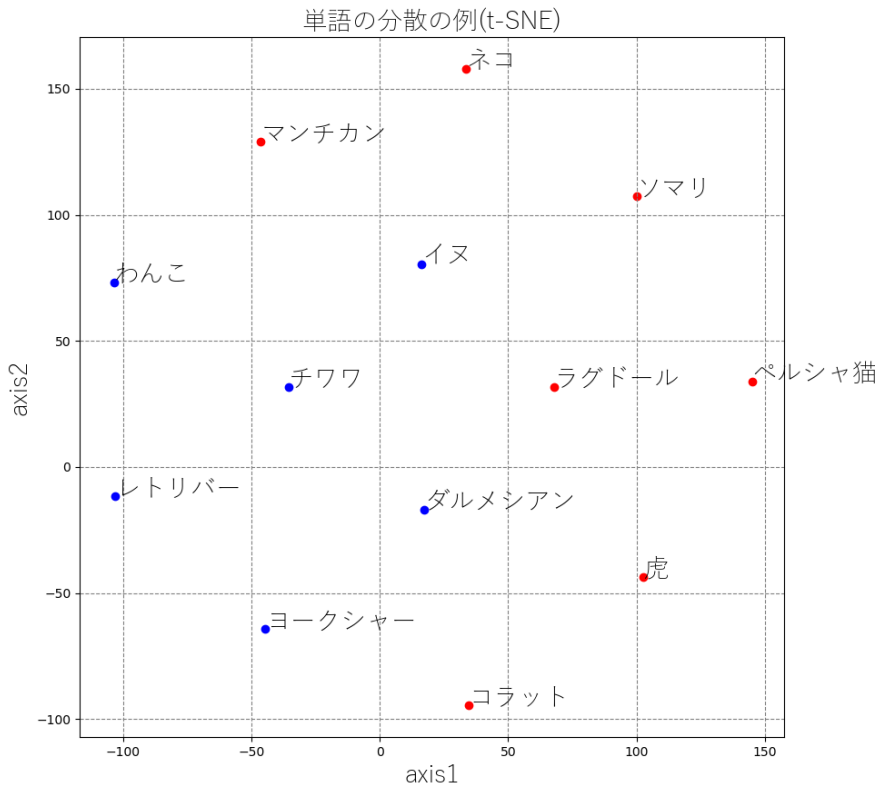

Figure 5:　単語分散の例 (t-SNE(t-Distributed Stochastic Neighbor Embedding(Maaten and G. Hinton 2008)) を用いて二次元平面に描画)

4.4 関連研究

単語分散を得るための手法としては、SVD (特異値分解 A singularly valuable decomposition(Kalman 1996)) や Word2Vec (Mikolov, Sutskever, K. Chen, Greg S Corrado, et al. 2013) や Glove (Pennington, Socher, and Manning 2014)、fasttext (Bojanowski et al. 2017) といった手法が有名だ。また昨今、NLP では文単位での解析が多いこと、文全体の意味も考慮したほうが良いというモチベーションから、単語分散のみならず、文ごとの関係も考慮してベクトルを生成する手法が提案されている。その代表例が、ELMo(Embeddings from Language Models Peters et al. 2018)、BERT(Bidirectional Encoder Representations from Transformers Devlin et al. 2018) といった深層学習のモデルであり、昨今の様々な NLP のタスクで SOTA を達成している。　また極性判定やカテゴリ分類において最近では画像認識の分野で広く使われている CNN を用いた研究も盛んであり、本研究ではそのうちの CNN と RNN (のうちの LSTM) を用いたモデルを用いて極性判定を実験する。

4.4.1 Skip-gram

Skip-gram (Mikolov, Sutskever, K. Chen, Gregory S. Corrado, et al. 2013) のアルゴリズムは以下 (4.4.1) のとおりである。[14]

Skip-gram のアルゴリズム

1. 正のサンプルとして、ターゲットの単語とその周辺の単語を取り出す。
2. 負のサンプルとして、単語辞書の中からランダムにサンプルされた単語を取り出す。
3. ロジスティックス回帰を用いてこの 2 つのサンプルを区別できるようにネットワークを訓練する。
4. ネットワークの重みを単語埋め込みとみなす。

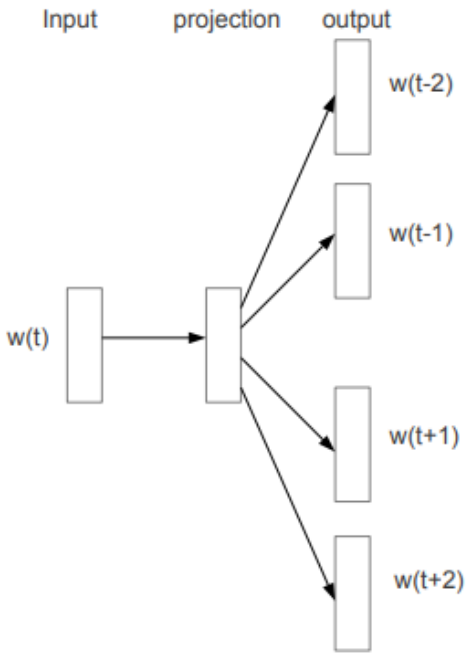

Figure 6: Skip-gram は文中におけるある単語の周辺単語を予測する (w(t) は t 番目の単語を示す。) (Mikolov, Sutskever, K. Chen, Greg S Corrado, et al. 2013 より)

[14]計算の都合上、辞書全体の単語を取り上げることが不可能なため、ネガティブサンプリングを行っている。またこのサンプリングは均一ではなく、高頻度な単語は程よく省かれるようになっている。(Mikolov, Sutskever, K. Chen, Greg S Corrado, et al. 2013)

Table 7: fasttext の Skip-gram を用いた単語分散獲得学習のパラメータ

パラメータ名	説明
許容最低語彙頻度	語彙として認める単語の頻度。 これを下回る単語は頻度の少ない単語として学習の対象としない。
学習係数	目的関数 Adagrad の学習係数。
学習係数向上率	学習率の更新率、単語がこの数だけ訓練されると更新される。
epoch 数	語彙の数 に対して何倍訓練を行うかを決定する。
ネガティブサンプリング数	学習ごとに負のサンプルをどのくらい抽出するか。
ウィンドウサイズ	アルゴリズムで説明した m の値
損失関数	損失関数
dim	埋め込みベクトルの次元数

ここで fasttext で用いられている subword との関連について説明する。まず Skip-gram の損失関数を以下の条件のもとで示すと以下のようになる。

但しこの式はある単語に対して一単語を予測する多クラス分類問題となっているが、実装上は 2 クラス分類へ変換されている。今回は簡単のためもとの多クラス分類問題のまま説明を進める。

1. Skip-gram で予測する単語はある単語の前後一単語のみ。
2. 単語を w_i 、コーパスを $[w_1, \cdots, w_T]$ とする。
3. ネガティブサンプルの手続きを省く。
4. 語彙数は W とする。

$$
\begin{aligned}
L &= -\frac{1}{T}\Sigma_{t=1}^{T}(logP(w_{t-1}, w_{t+1}|w_t)) \\
&= -\frac{1}{T}\Sigma_{t=1}^{T}(logP(w_{t-1}|w_t) + logP(w_{t+1}|w_t))
\end{aligned} \tag{1}
$$

この際に通常の Skip-gram では $P(w_c|w_t)$ は以下の式で表される。

$$
P(w_c|w_t) = \frac{e^{s(w_t, w_c)}}{\Sigma_{j=1}^{W} e^{s(w_t, j)}} \tag{2}
$$

問題はこの内の関数 s だ。この関数は 2 つの単語を引き取って類似度のスコアを返す関数で、通常の Skip-gram ではそのまま 2 つの単語を独立の id が振られたものとして処理している。しかし subword を用いている fasttext ではこの関数の定義が異なっている。具体的には単語の文字的な n-gram を取り、その n-gram の集合 $\mathcal{G}_{w_t}$ のそれぞれについての埋め込みベクトル z_g と周辺単語の埋め込みベクトル v_{w_c} との類似度を内積として計算している。

具体的に示すと以下のようになる。

$$
s(w_t, w_c) = \Sigma_{g \in \mathcal{G}_{w_t}} z_g^T v_{w_c} \tag{3}
$$

文字的な n-gram の例

元の単語 : where
tri-gram を取った場合:
<wh, whe, her, ere, re>

4.4.2 CNN-LSTM

CNN-LSTM (Sainath et al. 2015) とは CNN と LSTM を組み合わせたニューラルネットワークだ。これと似たものに、LSTM-CNN というものがあるが、両者の違いは、入力からみて先に CNN 層 を通過するか、LSTM 層 を通過するかというものだ。また LSTM 層 は 双方向 LSTM や GLU(Chung et al. 2014) といった RNN をの派生ネットワークに置換されることがある。尚、CNN-LSTM はいくつか呼び名があり、参考文献

として挙げたもの (Sainath et al. 2015) には CLDNN という名称で呼ばれている。

[Figure 10] に概略を引用する。ここでは下から上へデータが流れていく形になっており、下の x_i は所謂単語を示すベクトルを表している。尚 linear Layer とは CNN からの出力の次元を削減するために用いられるレイヤーを示している。また本実験では LSTM layers は 双方向 LSTM を一層だけ用いている。

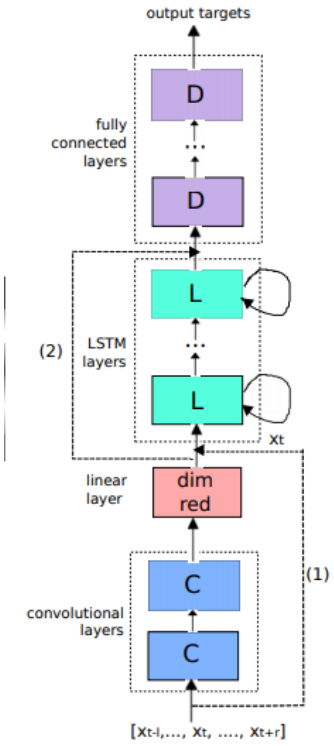

Figure 7: CLDNN の概略 (Sainath et al. 2015 より)

4.5 実験 1) 漢字かな問題に対する単語分散獲得

この実験では、日本語特有に存在する “漢字とかなによる同意表現の複数表記” を解消するための漢字 → かな変換を行い、それによって得られる性質の変化を調査する。

上記の調査で明らかになったように、日本語には同意でありながら様々な表現が存在している。その中でも比較的簡単に差がわかる・前処理が簡単であるものとして、“漢字とかな” について挙げることができる。例えば “寒い” という単語は “さむい”、“寒い” といった場合があるが、これらは単語的にはほとんど同じ意味を示す。また漢字とかなが入り交じることによって文字の種類が増加し、英語に比べて解析時の次元数が増大してしまう可能性が直ちにわかる。更に日本語のみならず英語を代表とした他国語をそれらの文字のまま併用し、それを当然のように会話に組み込んでいるという特徴から、日本語の文字種数を削減することは重要であると考えられる。そこで漢字をすべてかなに変換するという前処理を実験する。

しかしこの前処理を行う弊害として、例えば “すなわち”、“即ち”、“則ち”、“乃ち” といった微妙にニュアンスの異なる同音の単語がまとめられてしまうことによる影響をについて憂慮する必要があり、考察しなければならない。

4.5.1 実験概要

単語分散を得るためのコーパスとして Wikipedia から入手したコーパスを用いた。Wikipedia コーパスを選択した理由として、プライバシーや料金といったデータの入手難易度が低いこと、言語モデルを作成することを視野にいれた際に、百科事典的な特徴から大まかに日本語の語彙を網羅することが期待でき魅力的であることを挙げられる。

前処理として行う 漢字 → かな変換には MeCab の辞書を用いて行った。

実験に用いるモデルは、fasttext の Skip-gram だ (Bojanowski et al. 2017)。fasttext は Skip-gram の機構に subword という仕組みが追加されており、使わない場合よりも良い性能が得られることが知られている。

subword とは活用や語幹といった単位で単語を分割することで、例え単語が文字上一致しなくともその単語間の距離が近くなることを保証できるという利点が得られる。これは特に英語が、単語が小さな意味を持つ文字群に分割できることに大きく影響する。この利点は日本語にも応用可能であるという理屈としては、任意の国語辞典を開けばわかることだ。

Skip-gram はターゲットとなる単語からその周囲単語を予測する単語分散の獲得手法である。Skip-gram と、 Skip-gram と subword の関係の概要は 4.4.1 で説明する。

評価の対象は以下の 3 点についてだ。

- 語彙数の変化
  漢字 → かな変換によりどれだけ語彙を縮小させることが出来たのかを調査する。
- それぞれの、単語埋め込みベクトルの次元数と損失の変化
  それぞれの場合で、単語埋め込みベクトルの次元数に対して、fasttext の訓練後の損失がどの程度変化するのかを調査する。
- それぞれで得られた最良のモデルに対する、類似語の変化
  それぞれの場合で、“日本 (ニホン)” という単語に対してどのような類似単語が得られるのかを調査する。

実験上の固定されたパラメータを以下に示す。パラメータの詳細な意味は 4.4.1 で説明する。

subword の例

・英語の場合
  inspire → in・spire (中に+吹き込む)

・日本語の場合
  鶏肉 → 鶏・肉 (鶏 (の)+肉)

Table 8: fasttext を用いた単語分散獲得学習の共通パラメータ

パラメータ名	値

次ページに続く

前ページからの続き

パラメータ名	値
許容最低語彙頻度	5
学習係数	0.1
学習係数向上率	100
epoch 数	5
ネガティブサンプル数	5
ウィンドウサイズ	5

4.5.2 実験結果

実験結果を示す。

ここでいう次元数とは単語埋め込みベクトルの次元数 dim であり、default とは漢字かな入り混じり文、yomi とは漢字 → かな変換を行ったものを示す。

左のグラフは語彙数を縦軸にしており、default に対して yomi が少ないことを示している。右のグラフは 100、200、300 の次元でどのように fasttext の訓練後の loss が変化するのかを調べたものである。これを行った理由は、default と yomi の語彙数の変化に伴い適切な単語埋め込みの次元数が変化している可能性を考慮したためだ。

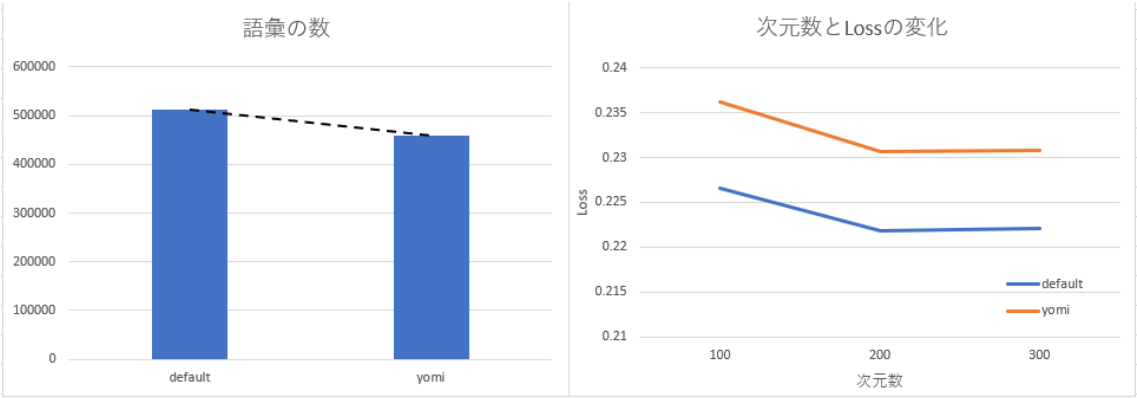

Figure 8: 漢字かな問題に対する単語分散獲得

1. 漢字かな入り混じり文 の類似単語

用いた単語埋め込みの次元数は 200 である。学習したモデルから、ターゲットである “日本” に対して類似している単語を上位から 10 個抽出した。

Table 9: 漢字かな入り混じり文 の類似単語

ターゲット	日本
類似単語	韓国
	米国
	台湾
	にっぽん
	中国
	日本さくらの会
	海外
	実業
	国内
	日本税理士会連合会

2. かなのみの文 の類似単語

用いた単語埋め込みの次元数は 200 である。学習したモデルから、m ターゲットである “ニホン” に対して類似している単語を上位から 10 個抽出した。

Table 10: かなのみの文 の類似単語

ターゲット	ニホン
類似単語	ニホンヤモリ
	ニホンバレ
	ニホンシカ
	ニホンウンソウ
	ニッポンザル
	ニホンズイセン
	ヒトツオボエ
	ゴジセイ
	ニホンカジョシュッパン
	ニホンドケン

4.5.3 考察

漢字 → かな変換によって語彙が 10%程度減少したことは確認できたが、損失は増加してしまったことがわかる。しかしいずれの場合でも次元数と損失の変化の外形は似ていることから Skip-gram の損失のみを見るならば変換前のテキストの方が良い単語埋め込みを獲得できていると考えられる。

また類似単語であるが、漢字かな入り混じり文は国として類似する単語を取り出していることがわかるのに対して、かなのみの文では 生物名 や、“日本晴れ” といった 慣用的な表現 を多く抽出している。このことから変換を行ったほうが、subword を活かすことが出来ていると考えられる。

これらのいずれが良いのかについては議論の余地があるだろうが、少なくとも汎用的な言語モデルを作成するならば前者の Skip-gram としての損失が小さい方を選択する方が良いと考えられる。

4.6 実験 2) 得られた単語分散を用いた極性判定

この実験では、4.5 で得られた単語分散を用いて極性判定を行うことで 2 つの単語分散の極性判定における性能を調査する。

一般に単語分散を獲得することで得られる言語モデルは極性判定やカテゴリ分類等に活用されることが多いが、今回は特に極性判定のうちの、陽性・中性・陰性の 3 値分類について挑戦する。3 値分類を選んだ理由は、データとして Twitter のデータを収集した際に、4.1 にあるように必ずしも陽性・陰性の 2 値を取

らなかったこと、5 値のようなより複雑な分類にすると、データのラベリングコストが高くなってしまうことを挙げる。

4.6.1 実験概要

用いた単語分散は 4.5 で得られた中で損失が最小であった 200 次元のものを用いた。極性判定のデータセットは 4.1 で抽出・編集したデータだ。抽出条件として、4.1.2 で得られた結果を用い、今回はこのいずれかに該当するものすべてを削除・編集した。

データ数は総データ数 1270 発話、この内ランダムに抽出した 10 %を学習に用いない検証データとした。

用いたモデルは CNN(Convolutional neural network (Fukushima 1980)/LeCun et al. 1999) と 双方向 LSTM(Bidirectional long short term memory(Schuster and Paliwal 1997/Graves, Fernández, and Schmidhuber 2005)) を合わせたもの (Sainath et al. 2015) であり、構成を以下に示す。

構成しているレイヤーの説明は 4.4.2 で行う。尚層数や各層の種類、ハイパーパラメータについては実験を繰り返す中で調節した。

Table 11: 実験に用いた CNN の概要

パラメータ (レイヤー)	値	補足
1 層	1 次元畳み込み	フィルターサイズ 64 / カーネルサイズ 3 / 活性化関数 elu
2 層	1 次元畳み込み	フィルターサイズ 64 / カーネルサイズ 3 / 活性化関数 elu
3 層	1 次元畳み込み	フィルターサイズ 64 / カーネルサイズ 3 / 活性化関数 relu
4 層	最大プーリング	プーリング幅 3
5 層	双方向 LSTM	隠れ層サイズ 256 / ドロップアウト率 0.2 / 再帰中のドロップアウト率 0.3
6 層	全結合層	ユニット数 256 / 活性化関数 sigmoid
7 層	ドロップアウト層	ドロップアウト数 0.25
8 層	全結合層	ユニット数 256 / 活性化関数 sigmoid
9 層	ドロップアウト層	ドロップアウト数 0.25
10 層	全結合層	ユニット数 256 / 活性化関数 sigmoid
1 層	ドロップアウト層	ドロップアウト数 0.25
12 層	全結合層	ユニット数 3 / 活性化関数 softmax
epoch	十分に学習できるまで	過学習が起きる直前の値を訓練後の精度とした
最適化関数	Adam	適当に調整した
損失関数	クロスエントロピー	

4.6.2 実験結果

以下のようにいずれの場合でも accuracy という面では若干の精度向上が見られた。しかし検証データの損失に関しては増大してしまっている。

Table 12: 得られた単語分散を用いた極性判定

	漢字かな	かなのみ
訓練データの損失	0.9523	0.7016
訓練データの accuracy	95.2%	98.2%
検証データの損失	1.204	2.096
検証データの accuracy	61.5%	64.8%

4.6.3 考察

ひらがなにすることでやや精度が向上したようにも見えるが、複数回実験をしたものの大きな違いが得られるような結果とはならなかった。この原因として、Wikipedia コーパスと収集したデータの距離が離れていることを考えることが出来る。

本実験でははは以下の表に示すように学習した語彙以外の単語が、いずれの場合でも 30%ほど、学習データに含まれてしまった。これは subword を用いての結果であるため、単語区切りやそれ以上の区切りのもので単語分散を学習した場合には、より語彙外の単語が増えてしまうことが想定できる。これに対処する方

法として、学習に用いるデータも合わせて fasttext で単語分散を得ることが提案できるが、Wikipedia コーパスに比べ学習データは極端に少ないため、2 つのデータを合わせてもそれらは語彙外の単語として切り捨てられてしまった。

以上のことから、前処理もさることながらより目的にあった密な (語彙数の増加よりもデータ数の増加が大きくなるような) データを効率よく大量に収集する必要があると考えられる。

また検証データに対する精度が向上しながらも損失が不安定になってしまうという問題が多く発生した。これは損失がクロスエントロピーを用いていることで、以下のような現象が起きていると考えられる。

クロスエントロピーを用いて損失が増大しまうシナリオ

真のラベルを $[1.0, 0.0]$ とする。出力をそれぞれ $[0.8, 0.2]$ 、 $[0.6, 0.4]$ とする
勿論いずれの場合においても正しく識別できている。
しかし真の分布 p(x) と 推定された分布 q(x) を用いてクロスエントロピーは以下のように定義されるものであるから、前者 (0.223) よりも後者 (0.510) の方が損失の値が大きくなってしまう。

$$cross\_entropy = -\Sigma_x p(x) \log q(x)$$

Table 13: 学習データ中の語彙外の単語数

	漢字かな	かなのみ
全語彙数	19265	20975
語彙外の単語数	6512	6453
割合	33.8%	30.1%

Table 14: 得られた単語分散を用いた極性判定 (Wikipedia + 学習データ)

	漢字かな	かなのみ
訓練データの損失	0.1161	0.1010
訓練データの accuracy	96.8%	96.9%
検証データの損失	1.7960	1.8166
検証データの accuracy	64.0%	64.7%

5 文抽出を念頭においた不均衡分散・サイズの分類問題

任意の文の入力を受け付ける際に、いくつかのある特定の内容の文が入力された場合のみ、何らかのイベントを発したいという状況について考える。このとき “任意の文” と “ある特定の内容” という領域の比を考えるといくつかのパターンが考えられる。例えば、“任意の文” が極性判定のようなネガティブ・ポジティブな文の集合であり、“ある特定の内容” がポジティブな文であったとき、これはネガティブな文とポジティブな文を区別するシンプルな 2 クラス分類問題と考えることができる。ここで用いる、シンプルな、という意味は、おおよそ 2 つのデータの自然言語空間上の分散、領域の大きさが一致していると考えられ、おおよそ同じくらいのデータサイズのサンプルを確保できるということだ。ところが、“任意の文” が例えば病院の診察記録であり、“ある特定の内容” が 1,000 万人に一人の発症率の難病、しかもそれを複数取り扱いたいと考えたとき、この問題は極めて難しいものとなる。これは n クラス分類問題でありながら、1 つのクラスが異様に全体データの領域を占め、そして残りの $n-1$ クラスが得られるデータのサンプル数が極端に少ない。こうなると通常のクラス分類ではうまく行くとは考えにくい。

本テーマでは、うまく行かないということを確かめるため、まずデータが充実している画像処理についてこの問題を考え、次に提案する手法である、点類似度を用いたクラス分類を実験し、その効果を確認する。

5.1 問題設定

3 つの問題設定で実験を行う。

一つは ImageNet という 2 万種類以上のラベルを持つ画像認識のデータセットを用いた 2 クラス分類で、猫の画像と犬の画像を分類する場合と、猫の画像ととランダムな画像を分類する場合、そしてそれぞれでデータ数に偏りをもたせた場合の精度比較する。本来ならば自然言語の分類問題として解きたい問題であ

るがデータセットを用意できなかったため、こちらで実験を行う。　最後に提案する点類似度を用いたクラス分類を行う。この提案手法は、与えられた文と判定したいクラスのテキストのサンプルデータすべてに対するの類似度をとり、その値群を考えることでその文がクラスに含まれているかを考えようというもので、値群を合計するのか、最大値を取るのかという 2 つの指標の下実験する。

5.2　実験) 画像タスクに置換した場合における一般的なクラス分類

ImageNet のデータを用いた画像タスクで、猫・犬分類と猫・ランダム画像でのクラス分類を行い、その精度の変化を実験する。

5.2.1　実験概要

ImageNet (Deng et al. 2009) とは 2 万件のラベルを持つ画像を合計で 1,500 万枚有しているデータベースである。

つまりここから得られる画像データセットを利用すれば、19,999:1 の比率のクラス分類を実験することができる。また深層学習の分野では積極的に画像認識で使われている技術が自然言語処理でも使われている [15] ことから、こちらで精度が出ていればそれを自然言語処理に転用することも容易であることが伺える。以上のことからこれは元問題の設定にそれなりに近い設定であると言えるだろう。

その上でデータの分散が異なると見られる犬とランダムな画像を相手として、猫の画像と分類する 2 クラス分類問題を実験する。

尚今回は比較のため、用いるモデルは統一している。そのモデルは AlexNet(Krizhevsky, Sutskever, and Geoffrey E Hinton 2012) を参考にした CNN (Convolutional Neural Network) であり、概要は以下の通りであり、詳細は 9.4.1 で述べる。

データは 28×28 の 3 チャンネル (rgb) の画像、データ数は猫・犬 (ランダム画像) で、その比率は 200:1000 / 400:800 / 600:600 / 800:400 である。検証データについてはいずれの場合でも 30:30 に統一した。

Table 15: 実験に用いた CNN の概要

パラメータ (レイヤー)	値	補足
1 層	2 次元畳み込み	フィルターサイズ 32 / カーネルサイズ 3×3 / 活性化関数 relu
2 層	2 次元畳み込み	フィルターサイズ 64 / カーネルサイズ 3×3 / 活性化関数 relu
3 層	最大プーリング	プーリング幅 2×2 / プーリング間のストライド 2
4 層	ドロップアウト層	ドロップアウト率 0.25
5 層	2 次元畳み込み	フィルターサイズ 128 / カーネルサイズ 2×2 / 活性化関数 relu
6 層	最大プーリング	プーリング幅 2×2 / プーリング間のストライド 2
7 層	2 次元畳み込み	フィルターサイズ 128 / カーネルサイズ 2×2 / 活性化関数 relu
8 層	最大プーリング	プーリング幅 2×2 / プーリング間のストライド 2
9 層	ドロップアウト層	ドロップアウト率 0.25
10 層	全結合層	ユニット数 1500 / 活性化関数 relu
11 層	ドロップアウト層	ドロップアウト数 0.5
12 層	全結合層	ユニット数 2 / 活性化関数 softmax
epoch	十分に学習できるまで	過学習が起きる直前の値を訓練後の精度とした
最適化関数	Adam	
損失関数	クロスエントロピーに重みを付けたもの	重みはデータ数 x:y に対して y:x の比率

5.2.2　実験結果

図中の Train_acc は訓練データに対する accuracy、Val_acc は検証データに対する accuracy、Train_loss は訓練データに対する損失、Val_loss は検証データに対する loss だ。尚 accuracy が 0、或いは損失が 1 となっているのは学習率などを変更しても収束しなかったことを示している。

[15] 例えば最近では RNN(reccurent neural network (Jain and Medsker 1999)) で文章のベクトルを生成していたものと、画像認識分野で広く使われている CNN(convolutional network) を用いて同様のことを行う研究 (Elbayad, Besacier, and Verbeek 2018b) が流行している。

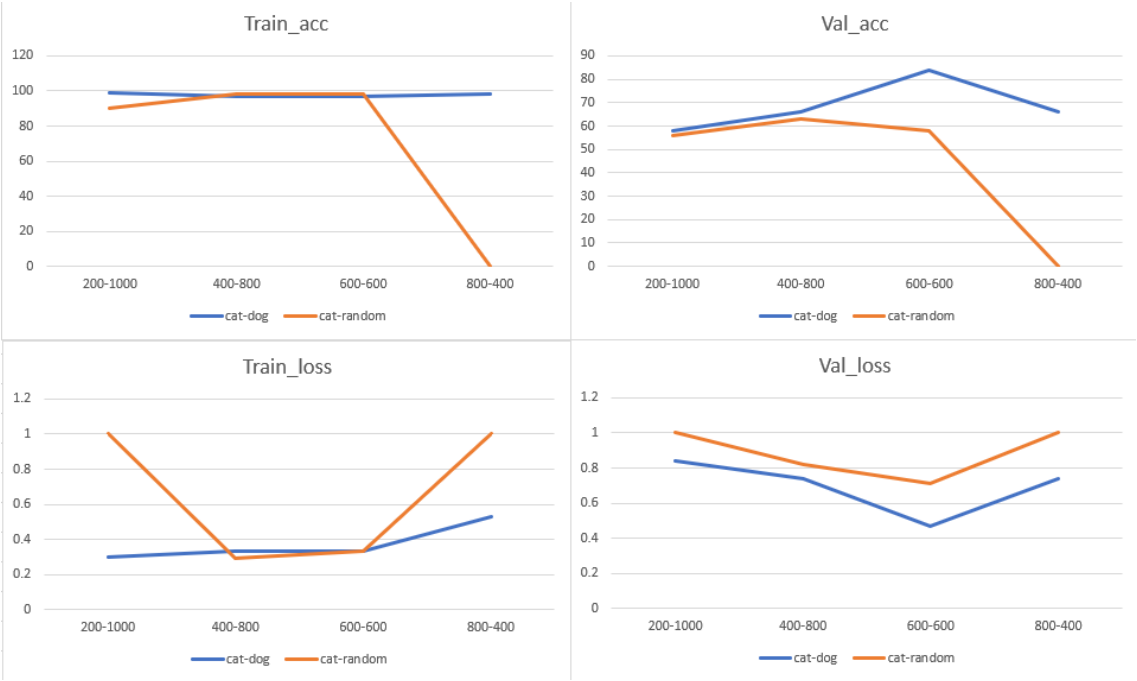

Figure 9: 画像タスクに置換した場合における一般的なクラス分類

5.2.3 考察

全体的にランダム画像とのクラス分類の方が精度が悪いとわかる。このことから、通常のクラス分類を転用してクラス分類を行うよりはそれにふさわしいモデルを作成した方が良いとわかる。

またランダム画像とのクラス分類に関しては、ランダム画像が多いほうが検証データに対する accuracy が向上するという予想があったが、ほとんど向上しないことがわかった。しかし犬画像との検証データに対する accuracy を比較すると、犬画像がデータ数が等しい場合を頂点として対称に精度が落ちているのに対して、ランダム画像に関しては 400:800 の時が最も精度が高くなっていることが興味深い。しかしいずれの場合でもデータの偏りが生じると損失は増加してしまう傾向にあるため、これが健全な学習結果であるすることは難しいだろう。

またより損失の重み付けを大きくした場合についても実験を行ったが、この場合には学習が荒れてしまい結果を得ることが出来なかった。

5.3 TODO: 実験) 自然言語処理の場合における点類似度を用いたクラス分類

5.3.1 問題設定

2 つの文章の類似度を計るための問題として MRPC というタスクが存在している。特定の文を取り出すということを考えた際にこれが適当な問題設定として考えられる。ところで 4 で議論したように日本語には表記揺れや漢字・かなの問題が存在しているため、例えば “おはようございます” といった入力に対して何らかの応答をしたいというシステムを考えたとき、その入力を特定することが一筋縄ではいかないと容易にわかる。本研究ではその問題に対処するため、ある程度の特定の文のバリエーションについての集合を用意して、その集合の各点からの距離を計ることで入力がその特定の文を示しているかを判定する。

5.3.2 実験概要

データとして特定の意味を表す文の集合を複数用意する。この文の集合群から任意の文をサンプリングし、同じ集合に属していれば類似した文、そうでなければ類似していない文として 2 値分類を行う。次に任意の入力文と学習に用いた文群との類似度を一組ずつ行い、要約統計を取ることでその入力分がどの文の集合に最も近いかを推定する。この手続きの中で、どの程度を境界としてどの文にも属していない、と判定するべきかも考察する。つまり実験の流れを示すと以下のようになる。

1. 文の類似度を学習する。

2. 入力文に対して学習データとの距離を計算する。

3. その文がどの文集合に近いのか、あるいはどの文集合にも近くないのかを要約統計を行い判定する。

使うモデルは BERT モデル であり、この言語モデルのファインチューニングを行うことによって 2 文の類似度を計ることを学習する。

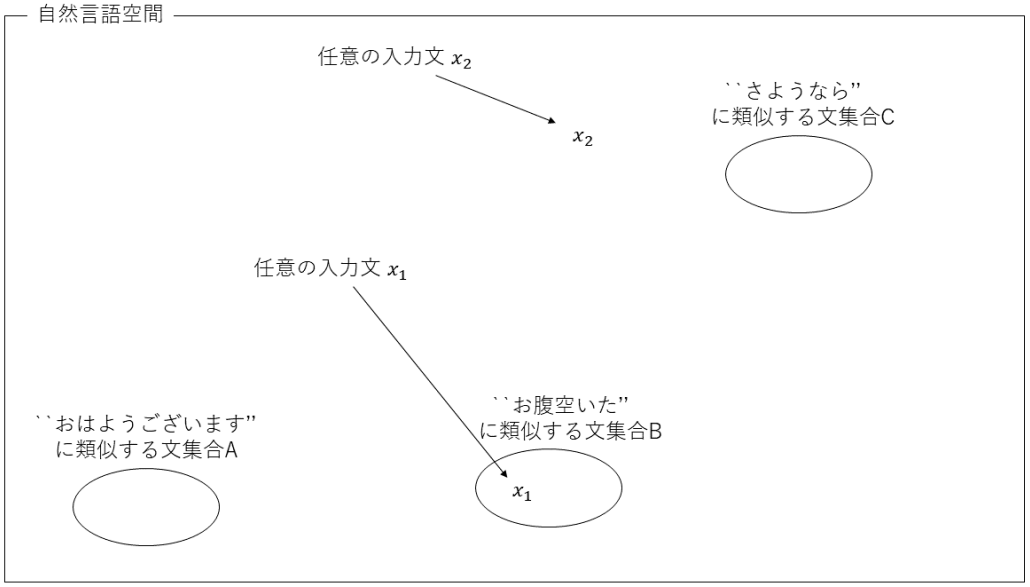

Figure 10: 自然言語処理の場合における点類似度を用いたクラス分類の概要図

5.3.3 実験

5.4 考察

後者のほうが拡張性があること、前者の場合に猫・犬よりも猫・ランダムのほうが精度が悪くなる傾向があることを指摘する。

6 機械翻訳システムを用いた対話モデル

6.1 問題設定

反射応答を行うシステムを作成するという問題について、機械翻訳の手法を用いることを提案、その手法として昨今機械翻訳の分野で SOTA を取っていた Transformer を用いることを実験し、その性能を考察する。

3 においてはカテゴリごとに別のモデルを作成することを提案しているが、本実験では十分なデータを入手できなかったため、利用可能なデータを集めたもので実験を行った。

6.2 実験) Seq2Seq Attention と Transformer の精度比較

6.2.1 実験概要

応答の中でも前後の文脈がなくともある程度意味が通じる反射応答に近い部分に適用されることを想定し、“入力文対出力文が一対一” であるようなシステムを考える。このときこの問題は、一般の機械翻訳の形に落ち着けることが出来る。この仮定のもとで、古くから有力な機械翻訳手法であるとされる Sequence to Sequence モデル (Sutskever, Vinyals, and Le 2014) [16] に補正をかけた Sequence to Sequence Attention (Luong, Pham, and Manning 2015) と、最近提案され 8 で取り上げる BERT という言語モデルの主軸となるアイデアにもなっている Transformer (Vaswani et al. 2017) を用い、それらから得られる出力の性質について考察する。

データとして用いたものは、自作の 1k 程度の対話と 4.2 において対話破綻チャレンジ (東中, 船越, and 小林 2015) から得ることが出来た 15k 程度の対話を組み合わせたものだ。自作のデータを用いた理由としては、対話破綻チャレンジで得られたデータセットを確認したところ、期待する反射応答である、挨拶といった対話が不足していると考えられたことを挙げる。

6.2.2 実験結果

以下に Sequence to Sequence Attention によって得られた出力と、Transformer から得られた出力、そしてそれぞれの BLEU スコア (Bilingual Evaluation Understudy Score) を示す。BLEU スコアについては 9.5.3 にて紹介する。

Table 16: データセットの入力と出力の例

入力	出力
チーズ ケーキ は 好き です ね 。	チーズ ケーキ の 美味しい 店 を 知っ て い ます 。
お は よー 。 熱中 症 に 気 を つけ て 。	おはよう ござい ます 。
テレビ は 民主党 支持 が 多い です ね 。	そんな 事 は ない でしょ ?
プレイ は 普通 です 。	どんな スポーツ を さ れ ます か ?
晴れ の 日 は 嫌い ? 暑い ?	晴れ の 日 が いい です ね 。
です よ ね 。	はい 。
すごい . 子供 とか 泣く もん ね 。	打ち上げ花火 だっ たら 連発 で ガンガン 上がる の が 楽しい 。
お は よー ありがとう 。	最近 映画 見 まし た ?
晴れ です か 。	気分 が いい です ね 。
ポテトチップス は うす 塩 が 好き です ね 。	うす 塩 は 定番 です ね 。
こんにちは ー っ 。	元気 が いい です ね 。

Table 17: Sequence to Sequence Attention と Transformer によって得られた出力

Seq2Seq Attn	Transformer
チーズ ケーキ の 美味しい 店 を 知っ て い ます 。	チーズ ケーキ の 美味しい 店 を 知っ て い ます 。
漫画 は 読み ます か 。	おはよう ござい ます 。

次ページに続く

[16] 一対一な機械翻訳手法を一般に “Sequence to Sequence (Seq2Seq) モデル” と呼ぶこともあるが、今回指すそれはそのうちの手法の一つであるモデル (Sutskever, Vinyals, and Le 2014) を指す。

前ページからの続き

Seq2Seq Attn	Transformer
そんな 事 は ない でしょ ?	そんな 事 は ない でしょ ?
どんな スポーツ を さ れ ます か ?	どんな スポーツ を さ れ ます か ?
晴れ の 日 が いい です ね 。	晴れ の 日 が いい です ね 。
夏 って 感じ が し ます 。	私 も スポーツ が 好き です 。
なるほど 。	気温 は いい です ね 。
最近 映画 見 まし た ?	最近 映画 見 まし た ?
気分 が いい です ね 。	気分 が いい です ね 。
うす 塩 は 定番 です ね 。	うす 塩 は 定番 です ね 。
元気 が いい です ね 。	元気 が いい です ね 。

Table 18: 学習に用いたデータの BLEU スコア

	BLEU スコア
Seq2Seq Attn	66.92
Transformer	77.11

Table 19: 学習外のデータについての BLEU スコア

	BLEU スコア
Seq2Seq Attn	61.80
Transformer	64.33

6.2.3 考察

それぞれのモデルからの出力文そのものを眺めると、いずれも文法的に不自然でないテキストを出力していることがわかる。しかしおおよそ短文としては成立している一方で文脈の考慮という点では今ひとつという出力が見られることがわかる。

しかし教師データである入力と出力がそもそも文脈上でのみ成り立っているものも含まれていることがわかるため、この点を考慮すればおおよそ期待通りの学習が出来たと考えている。

また本テーマでは翻訳とは違って単語対単語の直接的なつながりが比較的薄く、機械翻訳よりも精度が落ちるのではないかという予測があったものの、Transformer に組み込まれている単語間の関係を示す Positional Encoding が効いているおかげか単語対単語の対応ではない (n-gram 的な精度 = BLEU が高い) 学習が出来ていると考えられる。(実際に Positional Encoding を削除した場合で実験を行った際には Sequence to Sequence Attention よりも精度が悪くなってしまった。)

訓練時間については Sequence to Sequence Attention よりも Transformer のほうが圧倒的に早かった。これはまず Transformer が RNN を用いていないという影響が大きいと考えられる。しかしそれであったとしても、epoch 数が前者は 700 程度必要であったのに対して、後者は 60 程度で収まっているという点が興味深く感じられた。こちらの理由に関しては、Attention 機構と RNN の機構を組み合わせることでモデルが比較的に大きくなってしまったというということが考えられるが、確証のある説を提示することは出来なかった。尚具体的には以下の自宅環境で実験を行った際に、前者は 60 分、後者は 25 分ほどで学習することが出来た。[17]

Table 20: 実験環境

OS	Windows 10 Education
CPU	Intel(R) Core(TM) i7-6700 CPU @ 3.40GHz
RAM	16.0GB
GPU	NVIDIA GeForce GTX 1080
Python	3.6.5

[17] 自宅環境で実験を行った理由は、研究室の計算資源よりもこちらのほうが計算速度が早かったためだ。

また日本語特有な出力について報告する。まず活用の問題である。日本語には多彩な活用があるが、それぞれを別の単語として語彙と認めてしまっているため、人間が少し推論をすれば意味を理解することが出来るが、正解とは言えない出力が見られた。次に未知語についての問題である。特に出現頻度が少ない単語については語彙に含めないことが一般的な NLP の処理として挙げられれ、稀に入出力の一部が未知語を表す “<unk>” となってしまうことがある。ところが日本語においてこの未知語は稀に無視しても問題がない場合がある。これは日本語特有の文の省略法によるものと考えられる。これは解析上では問題となるが、出力においては他言語に比べ自然さについての基準を下げられるものとと考えている。

活用によって不自然な文になってしまった場合

お前 は 気が 合わ ます ね

合う + ます という文を生成したいと容易に考えられる。しかし “合う” の連用形は “合い” となるのに対して、“合わ” は未然形だ。一般に現代語の動詞は 5 段の活用形を持っており、これを別々のものとして語彙に含めるのは語彙の増加を助長するものと言える。またこのような活用の不一致が生じることを考えるならば、単語分割で語彙を作成しているという前提においては、一旦活用を終止形にしてしまい、出力結果からルールベースを用いるなどして動詞を活用させるといったの処理を想像することが出来る。

未知語 “<unk>” が省略できる場合

<unk> <unk> 大丈夫 ?

この場合、 “<unk>” を省いても話の本旨である “(文脈上の何かに対するものが) 大丈夫か尋ねる” という意味は伝えることが出来ると考えられる。

7　文のスタイル変換

日本語は英語と比較してペルソナに伴う語尾などの言葉遣いの変化が顕著である。これは日本語を対象とした統計・機械学習を行う際に、英語で用いられる手法を直ちに用いることができるか、という点で議論が生じる。その意味で日本語の文に対して何らかのスタイルを付与するという手法について既存の英語で用いられている手法と、2 つの提案手法を用いて実験する。

7.1　関連研究

スタイルを変換するという問題に対して、画像認識では VAE(Variational autoencoder(Kingma and Welling 2013)) や GAN (Generative Adversarial Network (Goodfellow, Pouget-Abadie, et al. 2014)) が提案されており、いずれも様々な派生が研究されている。NLP の舞台でも同様の試みが行われているが、現在文を変換するというタスクに対しては特に、VAE を用いた研究が盛んである。例としては Toward Controlled Generation of Text(Hu et al. 2017) や、Sequence to Better Sequence(Mueller, Gifford, and Jaakkola 2017) や Style Transfer from Non-Parallel Text by Cross-Alignment (Shen et al. 2017) があり、これらは非並行、つまり必ずしも元の文とスタイルが付与された文が対になっている必要がないという点で優れている。

しかしこれらが議論しているスタイルとは日本語で用いられるようなペルソナを象るようなものではなく、むしろ極性や単語並び替えといった議論に集中している。唯一 Sequence to Better Sequence に関してはシェークスピアの作品と現在の言葉との変換を行っているため、本実験ではこれを用いて実験を行う。

7.2　問題設定

日本語での書き言葉 → 話し言葉変換を行うことを問題として取り上げた。これは、元の問題である対話エージェントが何らかの人型、ないし何らかのキャラクタを持つことを想定したこと、学習させるための研究資材として個人で収集できる範囲の有効なデータが、Wikipedia や青空文庫 [18] の書籍といった言った比較的書き言葉に近いテキストを用いることになるだろうと考えていること、文章生成の段階では書き言葉の方がテキストの情報を正規化して持っているのではないかという予想があったことのためだ。

この実験における書き言葉と話し言葉の例を以下に引用する。これらのデータはデータセットを入手することが出来なかったため、自作のものを用いた。データの総数は約 300 と小さめのデータセットだ。

ここで並行なデータを用意していることについて触れる。まず第一にデータを作成する際に片方のデータセットを作成した後、も片方のデータセットを作成した方が効率が良かったことを挙げることができる。また今回議論するスタイル変換は上記の例のように極めて変化が乏しいものである。そのためこの変換を “特定の条件で特定の単語を置換する” という問題として見直し、これを解く手法を提案したことを挙げることができる。以上のことから今回は並行なデータを用いて実験を行う。

Table 21: 文スタイル変換に用いる学習データ例

書き言葉	話し言葉
おはようございます。	おはよう。
明日も会社です。	明日も会社だ。
明日はゆっくりできそうです。	明日はゆっくりできそう。
きっとうまく行きますよ。	きっとうまく行くよ。

7.3　実験) 書き言葉→話し言葉のスタイル変換

7.3.1　実験概要

Sequence to Better Sequence と Sequence to Better Sequence に Denoising autoencoder(Vincent et al. 2008) を加えたもの、CopyNet(Gu et al. 2016) を用いたものの 3 つについて同じデータセットで実験を行った。それぞれのモデルの説明は、9.6 で行う。

7.3.2　実験結果

以下に得られた結果を示す。学習精度については学習に用いたデータが少ないため議論できない。尚、S2BS は Sequence to Better Sequence、S2BS with DAE は Sequence to Better Sequence に Denoising autoencoder を

[18] https://www.aozora.gr.jp/

加えたものを示す。

Table 22: 文スタイル変換の実験結果

実装	入力	出力
S2BS	おはようございます。	おはよう。
	応援する。	応援してる。
	今日は寒かった。	今日は寒かった。
	夕飯は？	夕飯はどうしようか？
	早く寝たい。	お風呂に入ろう。
	何か不安だなぁ。	何か口に入れてはどうでしようか
S2BS with DAE	S2BS と同じ結果が得られた	
CopyNet	おはようございます。	おはよう。
	今日は良い天気ですね。	今日は良い天気。
	こんにちは。	こんにちは。
	頑張るぞい！	頑張るぞい！
	進捗どうですか？	進捗どう？

7.3.3 考察

本実験結果は学習データが極めて少ないものの、データが極めてノイズが少ないこともあってか、ある程度求めていた出力を得ることが出来たと考えている。しかし Sequence to Better Sequence に Denoising autoencoder がどのような影響を示すのかを確認することは出来なかった。ただ学習を行って見た感想としては、 Denoising autoencoder を加えた方が学習が難しくなっているように感じたが、これは入力の一部をマスクしている性質上当然とも言えるだろう。

Sequence to Better Sequence の出力例の後者 2 つについては非常に興味深い出力と言える。勿論学習データにはこのような変換を指定していないが、このように入力文に対して飛躍した文が生成されている。しかしこの入出力には全く相関がないとは言い切れないところが面白い。例えば、“早く寝たい” から “お風呂に入ろう” という変換は、“寝る前に風呂に入る” という学習内容に含まれない “生活習慣” を学習しているとも取れるもので、つまりは 4 で話題としたような言外の知識が学習されている可能性を示唆しているとも考えられる。

CopyNet に関しては、“単語の置き換えをする” という目的を達成しているということが、適当な入力をしてもそれが変換されずに飛ばされているという点から推測できる。このことから、任意の別のペルソナを持つ発言や文章を収集し、それぞれを学習データとしたとき、ペルソナを象りやすい単語を抽出することができるのではないかという可能性を想像することができる。またデータ数が少ないという問題を考慮したとき、CopyNet はその構造上未知語への対応が比較的容易であるため、データを集めることが困難な個人の研究者にとっては有効な手段であると考えられる。

8 CoLA タスクを応用した対話システムのエラー検知

8.1 問題設定

機械学習を用いて文章を作成する手続きの中では、ほぼ間違いなく “不自然な文” が生成されてしまう。ここで定義する不自然な文とは、語順や文法、そして意味を挙げることが出来る。

英語を用いた研究では、不自然な文と自然な文とを識別するためのタスクとして CoLA (The Corpus of Linguistic Acceptabillity (Warstadt, Singh, and Bowman 2018)) と呼ばれるものが存在している。

本テーマでは、機械学習モデルから生成された日本語の文を、不自然な文と自然な文とに識別するという問題設定を行い、実際に 6 から生成されたテキストに対してラベリングを行い、その識別を行う。

8.2 実験) 対話システムのエラー検知

8.2.1 実験概要

言語モデルである BERT (Devlin et al. 2018) を用いて自作のデータセットを用いて自然な文と自然でない文を判定するファインチューニングを行った。

データは 844 の文とそのラベルであり、ラベルは 0(不自然な文), 1(自然な文) の 2 値である。6 で期待できるデータに対して少ないように考えられるが、これは 6 の出力が想定以上に良いものであり、不自然な文を十分に用意できなかったためだ。またそのうちの 10% を検証データとして用いた。　以下に実験に用いたデータの例を示す。

Table 23: 対話システムのエラー検知のデータ例

ラベル	文
0	塩は強めです。
1	コーヒーとか?
0	の袋にてます。
1	このあたりの好みは似ていますね。
1	うす。

8.2.2 実験結果

ここでの loss と accuracy は検証データに対するものである。

Table 24: 対話システムのエラー検知の実験結果

	epoch	accuracy	loss
最も accuray が高いもの	30	0.702381	2.375742
最も loss が低いもの	3	0.619048	0.712082

以下に epoch と accuracy, loss についてのグラフを示す。

8.2.3 考察

元の CoLA タスクでの精度が 60% 前後であったのに対してより良い結果が得られたため、この結果が満足できるものであると考えられる。しかし loss と accuracy、精度の関係について疑問が今後の課題として残った。実験結果で示したように、accuracy と loss の最良値をとる epoch 数は一致しておらず、グラフとして見ても理想的な外形を得ることが出来なかった。検証データを取り替えてもこれ以上の結果を得ることが出来なかったため、データを増やすか、いずれかの値を “精度” の判断基準として採用する必要があると考えられる。“一般的には”[19] loss を判断基準として用いることが多いため、こちらを採用するべきだと予測できる。

[19]厳密にどちらかと明言された文書を見つけることが出来なかった他、ここ (https://stats.stackexchange.com/questions/258166) に興味深い議論があるように、頭ごなしに loss のみを観測して過剰適合かどうかを判定するのは早計であると考えたため、“一般的” という表現を用いた。

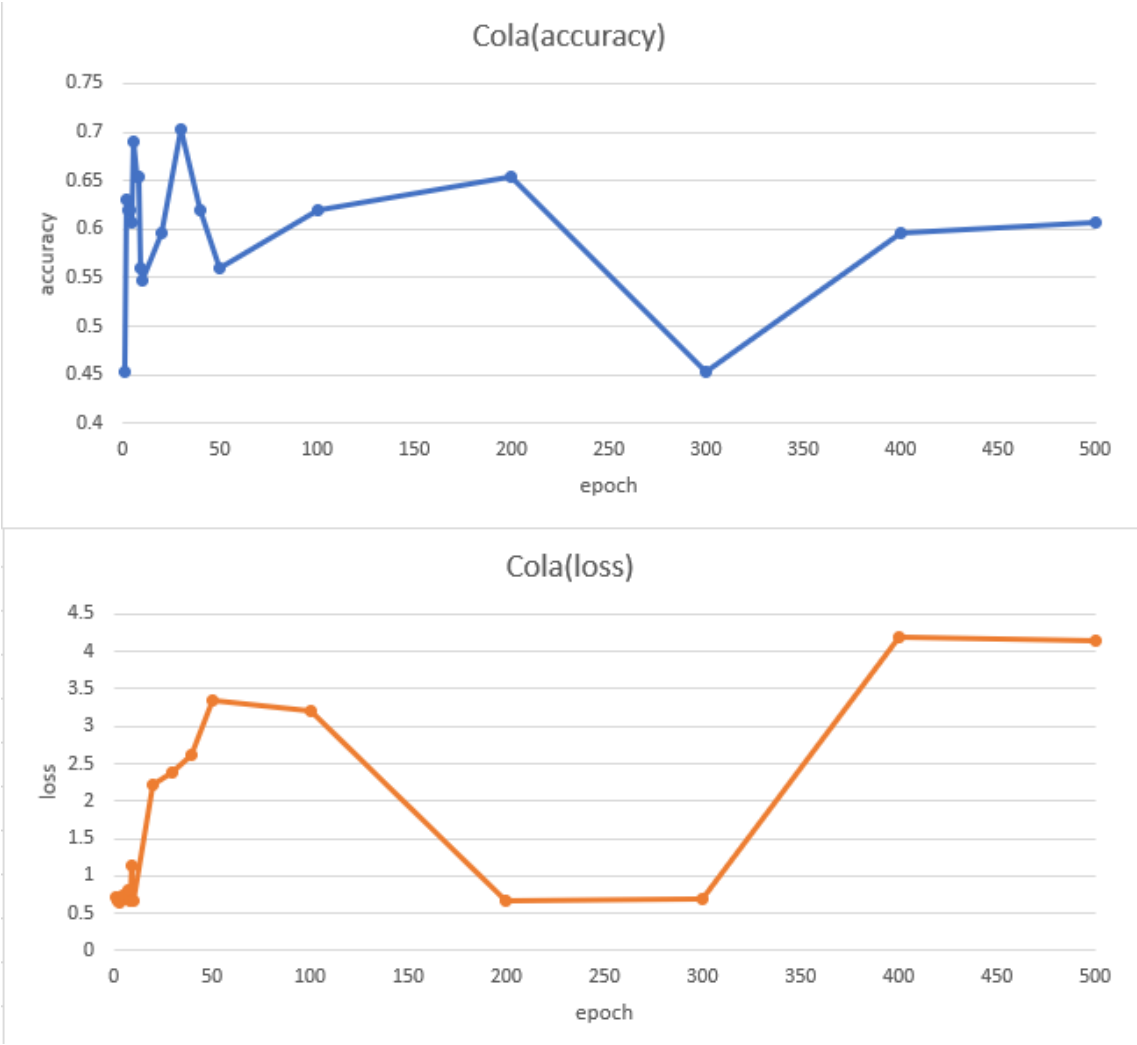

Figure 11: 対話システムのエラー検知の実験結果 における epoch と 精度の変化

9 付録

この章ではこれまでの章で登場したモデル・論文・用語などについての補足説明を行う。更に深い説明に関しては参考文献を参照していただきたい。

9.1 対話システムの関連研究

9.1.1 NLP における相互参照

NLP において相互参照 (coreference resolution) とは、具体的に以下の例を考えることが出来る。人間はこの代替を容易に補完することが出来るが、一般に機械がこれを理解することは難しいことが知られている。また日本語での対話の場合、このような指示語は省略される場合が英語等に対して比較的に多いため、指示語を補完する以上のものを要求されることになると考えられる。

この分野の研究としては、Stanford Core NLP[20] を挙げることが出来るが、これは対話データに対しては良い結果が得られないことが Gunrock で言及されている。Gunrock では対話データに対してアノテーションを行ったものを用いて、該当部の付加情報として持つということを行っている。

NLP における相互参照の例

HumanA: X さんは機械学習について研究しているらしい
HumanB: 彼 は そんな 研究に興味があるんだ。 それ にしては研究室があっていないような気がするね。
HumanA: そう かもしれないね。何かあったのかもしれない。

下線部に注目すると、1 つ目から順に “X さん”、“機械学習”、“機械学習を研究していること”、“研究室があっていないこと” を指し示している。

9.2 深層学習の基礎知識

9.2.1 CNN

CNN(Convolutional neural network (Fukushima 1980)/LeCun et al. 1999) とは格子型のトポロジーを持つデータを解析するためのニューラルネットワークの一種だ。例えばそれは二次元な (RGB 8bit) デジタル画像 ($\mathbb{N}'^{x\times y}$ $\mathbb{N}' \in \{0, 2, \ldots, 254\}$)、或いは一単語をベクトル $x_i \in \mathbb{R}^{x\times 1}$ としたときの文章ベクトル $[x_1, \ldots, x_n] \in \mathbb{R}^{x\times n}$ を挙げることが出来る。

ここでいう畳み込みとは、線形変換の一種であり、以下の式 (この式は一次畳み込み) で表されるものを指している。

$$\begin{aligned} s(t) &= \int x(a)w(t-a)da \\ &= (x \star w)(t) \end{aligned} \tag{4}$$

CNN において x はデータの多次元配列を示しており、w はカーネルと呼ばれる学習されたパラメータの多次元配列を示すこととなる。

より実計算に近づけるため、これを 2 次元実空間上の離散問題として再定義すると、多次元配列を $I \in \mathbb{R}^{M\times N}$ 、カーネルを $K \in \mathbb{R}^{m\times n}$ 、バイアスを $B \in \mathbb{R}^{M-m\times N-n}$ 、出力を $S \in \mathbb{R}^{M-m\times N-n}$ として以下の式に変換される。(正確にはこの手続きの途中で畳み込みを相互相関 (Cross-correlation) と解釈し直している (t-a → t + a) が、 a, b の符号を反転させれば同様のことが言える。)

$$\begin{aligned} S_{i,j} &= B + \Sigma_{a=0}^{m-1}\Sigma_{b=0}^{n-1} I_{i,j}K_{i+a,j+b} \\ &= B + \Sigma_{a=0}^{m-1}\Sigma_{b=0}^{n-1} K_{i,j}I_{i+a,j+b} \end{aligned} \tag{5}$$

[20] https://stanfordnlp.github.io/CoreNLP/coref.html

このとき誤差関数を E としてパラメータ K, B について勾配を求めると、

$$\begin{aligned}\frac{\partial E}{\partial K} &= \Sigma_{i=0}^{M-m}\Sigma_{j=0}^{N-n}\frac{\partial E}{\partial S_{i,j}}\frac{\partial S_{i,j}}{\partial K_{i,j}} \\ &= \Sigma_{i=0}^{M-m}\Sigma_{j=0}^{N-n}\frac{\partial E}{\partial S_{i,j}}I_{i+a,j+b} \qquad (6)\\ \frac{\partial E}{\partial B} &= \Sigma_{i=0}^{M-m}\Sigma_{j=0}^{N-n}\frac{\partial E}{\partial S_{i,j}}\frac{\partial S_{i,j}}{\partial B_{i,j}} \\ &= \Sigma_{i=0}^{M-m}\Sigma_{j=0}^{N-n}\frac{\partial E}{\partial S_{i,j}} \qquad (7)\end{aligned}$$

この勾配を用いて逆伝搬を行い学習を行うことになる。例として SGD(Stochastic gradient decent) を用いて K を更新する手続きを考えると 、ステップ数を t 、学習係数を η として以下のようになる。

$$K^{(t+1)} = K^{(t)} - \eta\frac{\partial E}{\partial K^{(t)}} \qquad (8)$$

また一般に CNN は一般に活性化関数についてもまとめられることがあり、本研究でもこの 2 つをまとめている。活性化の関数については sigmoid 関数や relu 関数、elu 関数などがある。

尚本研究の CNN の文脈で登場するフィルターサイズとは以上のシーケンスを行う数で、例えばある画像に対しフィルターサイズ 128 の畳み込みを行う、というのは 128 枚の同じサイズの出力結果 (それぞれの出力結果が等しいとは限らない) が得られるということを示している。

9.2.2 活性化関数

活性化関数は非線形関数の一種で、例えば sigmoid(logistic sigmoid) 関数、relu(rectified linear unit) 関数 (Nair and Geoffrey E. Hinton 2010)、elu(exponential linear unit) 関数 (Clevert, Unterthiner, and Hochreiter 2015) を挙げることが出来る。

$$\begin{aligned}sigmoid(z) &= \frac{1}{1+e^{-z}} \qquad (9)\\ relu(z) &= \max\{0, z\} \qquad (10)\\ elu(z) &= \begin{cases} z & (z < 0) \\ e^x - 1 & (z \leq 0) \end{cases} \qquad (11)\end{aligned}$$

9.2.3 最大プーリング

最大プーリング (層) とはプーリング (層) の一種である。プーリングを行うプーリング関数とは特定の場所のネットワークの出力を周辺の出力を最大値や平均値といった要約統計量で置き換える処理を行う関数だ。例えば最大プーリング層は最大値を要約統計量として用いるプーリング関数を採用したプーリング層で、平均プーリング層は平均値を要約統計量として用いるプーリング関数を採用したプーリング層だ。

更にこれを発展させるものとして、ストライドと呼ばれる場所の特定の指定幅を調節する (標準では 1 、このとき特定の場所は前の特定の場所の一つ隣を示す。) 場合もある。

9.2.4 RNN

RNN(reccurent neural network (Jain and Medsker 1999)) とは時系列データを解析するためのニューラルネットワークの一種だ。これを簡潔に紹介するために、パラメータ θ を持ち $t-1$ の状態 h を外部入力 x_t を伴って t へ遷移させる f という関数を用いる、以下のシステムを考える。

$$h^{(t)} = f(h^{(t-1)}; x^{(t)}; \theta) \qquad (12)$$

一般に RNN ではこのシステムをモデル化していると言える。つまり RNN は入力データ $\{x^1, \cdots, x^\tau\}$ をある関数 f を繰り返し適用しているとも言い直すことが出来る。

また本研究中で登場した双方向 LSTM はこの入力データを反転したもの $\{x^\tau, \cdots, x^1\}$ も考慮しようというモチベーションに基づいている。[Figure 12]

もう少し数学を用いて説明するため、標準的な RNN の構造を示す [Figure 11]。　右は左を時系列的に展開した図だ。各変数は以下のような意味を持っている。

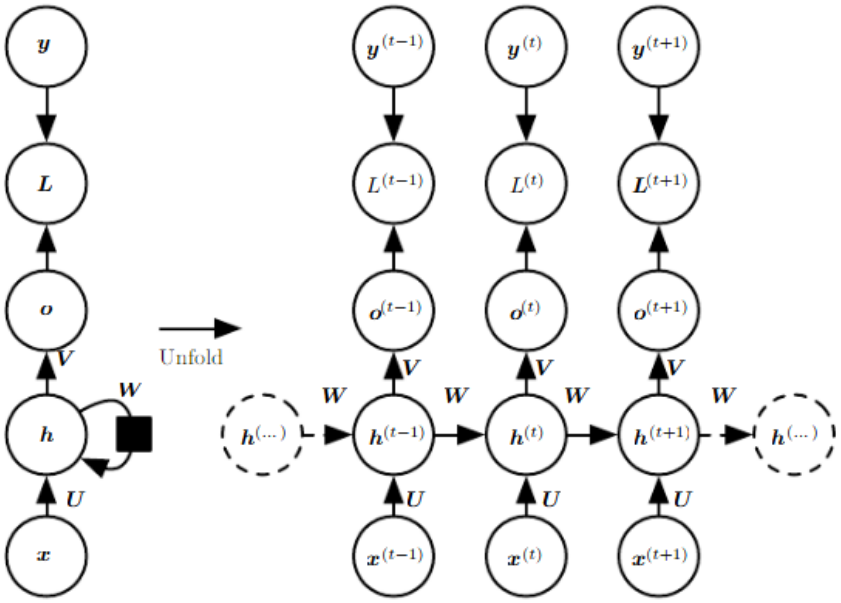

Figure 12: 標準的な RNN の構造 (Goodfellow, Bengio, and Courville 2016a より)

- x 外部入力で、一般にベクトルが用いられる。
- U 外部入力を状態へ反映させるための重み行列を示している。
- h 状態を示しており、一般にベクトルが用いられる。隠れ状態とも呼ばれる。
- W 隠れ状態を更新する重み行列を示している。
- V 隠れ状態から出力を計算する重み行列を示している。
- o 出力で、一般にベクトルが用いられる。
- L 損失で、モデルからの出力と正解との差 (厳密な距離である必要はない) を示す。
- y 正解を示しており、o と同じもの (例えばベクトル) だ。

この構造についていくつかの式を定義することが出来る。尚この式で登場する b, c はバイアスベクトルを示している。

$$a^{(t)} = b + Wh^{(t-1)} + Ux^{(t)} \tag{13}$$
$$h^{(t)} = tanh(a^{(t)}) \tag{14}$$
$$o^{(t)} = c + Vh^{(t)} \tag{15}$$
$$\hat{y}^{(t)} = softmax(o^{(t)}) \tag{16}$$

これらの式を繰り返し適用していくことで RNN は更新されていくことになる。
次に損失について求める。損失は外部入力と正解を用いて、次の式で求められる。尚、ここで言う $p_{model}(y^{(t)}|\{x^{(1)}, \cdots, x^{(t)}\})$ は 出力ベクトル $\hat{y}^{(t)}$ における $y^{(t)}$ の確率を考えているものとする。

$$L(x^{(1)}, \cdots, x^{(\tau)}, y^{(1)}, \cdots, y^{(\tau)}) = \Sigma_t L^{(t)} \tag{17}$$
$$= -\Sigma_t log p_{model}(y^{(t)}|\{x^{(1)}, \cdots, x^{(t)}\}) \tag{18}$$

勾配の計算については定義しなければならない要件が多いため、参考文献 (Goodfellow, Bengio, and Courville 2016a) を参照して頂きたい。

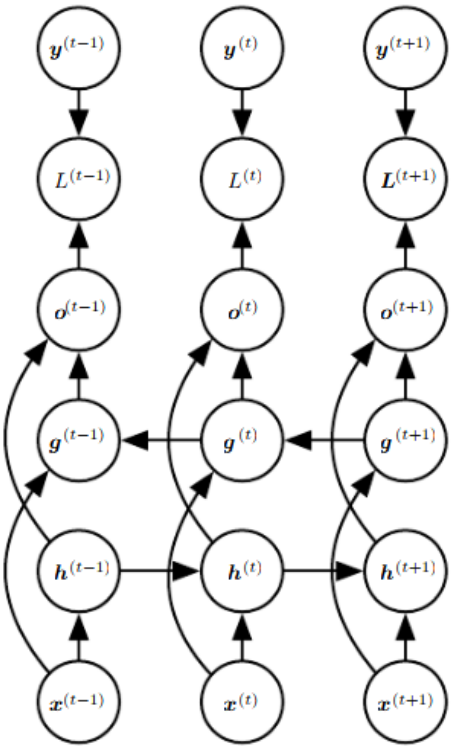

Figure 13: 双方向 RNN の構造 (Goodfellow, Bengio, and Courville 2016a より)

9.2.5 LSTM

LSTM(long short-term memory) とは RNN が長期的な系列の性質を保持しづらいという問題を解決する手法の一つ、自己ループ (内部回帰) を持つ “LSTM セル” という構造を加えた RNN の派生だ。LSTM セルの構造を [Figure 13] に示す。特に $\int$ のような記号はシグモイド関数を示しており、$+$ は加算、x は乗算を示している。

RNN との差異を明確にするため、h の更新式を示していく。まず LSTM セルの input について説明する。

input はバイアスベクトル b と入力に対する重み行列 U 、隠れ状態を更新する重み行列 W を用いて以下の式で求められる。

$$input_i^{(t)} = sigmoid(b_i + \Sigma_j U_{i,j} x_j^{(t)} + \Sigma_j W_{i,j} h_j^{(t-1)}) \tag{19}$$

input gate も同様に、バイアスベクトル b^g と入力に対する重み行列 U^g 、隠れ状態を更新する重み行列 W^g を用いて以下の式で求められる。

$$g_i^{(t)} = sigmoid(b_i^g + \Sigma_j U_{i,j}^g x_j^{(t)} + \Sigma_j W_{i,j}^g h_j^{(t-1)}) \tag{20}$$

以下 forget gate、 output gate も同様に、

$$f_i^{(t)} = sigmoid(b_i^f + \Sigma_j U_{i,j}^f x_j^{(t)} + \Sigma_j W_{i,j}^f h_j^{(t-1)}) \tag{21}$$

$$o_i^{(t)} = sigmoid(b_i^o + \Sigma_j U_{i,j}^o x_j^{(t)} + \Sigma_j W_{i,j}^o h_j^{(t-1)}) \tag{22}$$

以上の式から LSTM セルの状態 s についての更新式は以下のように書ける。

$$s_i^{(t)} = f_i^{(t)} s_i^{(t-1)} + g_i^{(t)} input_i^{(t)} \tag{23}$$

そしてこの LSTM セルの状態と output gate から LSTM セルの出力である $h_i^{(t)}$ が求められる。

$$h_i^{(t)} = tanh(s_i^t) o_i^{(t)} \tag{24}$$

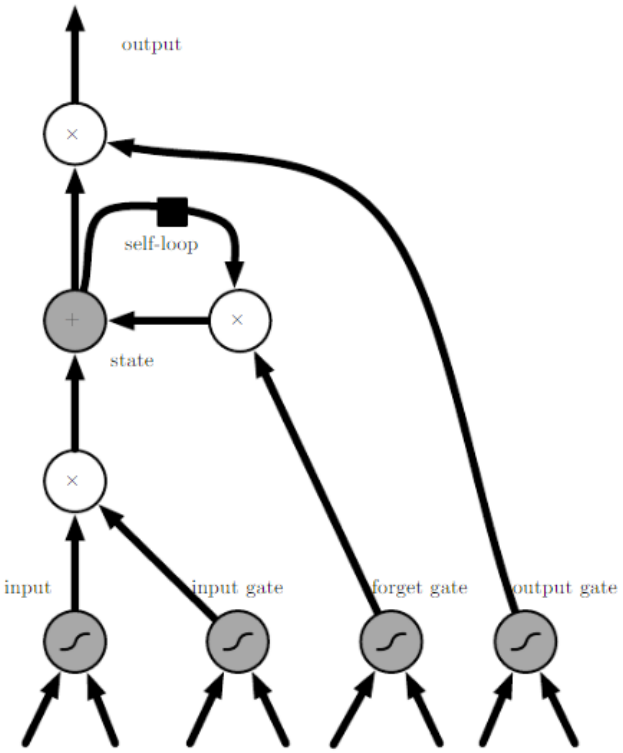

Figure 14: LSTM セルの構造 (Goodfellow, Bengio, and Courville 2016a より)

9.2.6 最適化関数

最適化関数とは、求められた損失に基づいてそれが小さくなるように重みを更新するための関数だ。例えば SGD (確率的勾配降下法 Stochastic Gradient Descent) や Adam といった種類が存在する。

SGD のアルゴリズム (Goodfellow, Bengio, and Courville 2016b より) を以下に示す。補足すると SGD の収束を保証する学習率の十分条件は以下のようになる。

$$\Sigma_{k=1}^{\infty}\epsilon_k = \infty \; and \; \Sigma_{k=1}^{\infty}\epsilon_k^2 < \infty \tag{25}$$

Algorithm 1 SGD の アルゴリズム

Require: 学習率 $\epsilon_1, \epsilon_2, \ldots$
Require: 初期パラメータ θ
  $k \leftarrow 1$
  while 終了条件を満たさない **do**
    訓練データ $x^{(1)}, \ldots, x^{(m)}$ と対応する目標 $y^{(i)}$ の m 個の事例を集めたミニバッチをサンプリングする
    勾配の推定値を計算する: $\hat{g} \leftarrow \frac{1}{m}\nabla_\theta \Sigma_i L(f(x^{(i)};\theta), y^{(i)})$
    パラメータの更新を行う: $\theta \leftarrow \theta - \epsilon_k \hat{g}$
    $k \leftarrow k + 1$
  end while

Adam のアルゴリズム (Goodfellow, Bengio, and Courville 2016b より) を以下に示す。

Algorithm 2 Adam の アルゴリズム

Require: ステップ幅 (推奨値 0.001)
Require: モーメントの推定を行うための指数減衰率 ρ_1 、$\rho_2 \in [0, 1)$ (推奨値 0.9 、0.999)
Require: 数値的な安定のために使われる小さい定数 δ (推奨値 10^{-8})
  一次・二次モーメントに関する初期値: $s = 0, r = 0$
  時間ステップの初期値: $t = 0$
  while 終了条件を満たさない **do**
    訓練データ $x^{(1)}, \ldots, x^{(m)}$ と対応する目標 $y^{(i)}$ の m 個の事例を集めたミニバッチをサンプリングする
    勾配の推定値を計算する: $\hat{g} \leftarrow \frac{1}{m}\nabla_\theta \Sigma_i L(f(x^{(i)};\theta), y^{(i)})$
    $t \leftarrow t + 1$
    バイアス付きの一次モーメントの推定値を更新する: $s \leftarrow \rho_1 s + (1 - \rho_1)g$
    バイアス付きの二次モーメントの推定値を更新する: $r \leftarrow \rho_2 r + (1 - \rho_2) g \odot g$
    一次モーメントのバイアスを修正する: $\hat{s} \leftarrow \frac{s}{1 - \rho_1^t}$
    二次モーメントのバイアスを修正する: $\hat{r} \leftarrow \frac{r}{1 - \rho_2^t}$
    更新量を計算する: $\Delta\theta = -\epsilon \frac{\hat{s}}{\sqrt{\hat{r}} + \delta}$ (この計算は要素ごとに計算される)
    更新量を適用する: $\theta \leftarrow \theta + \Delta\theta$
  end while

9.2.7 VAE

VAE(Variational autoencoder(Kingma and Welling 2013)) は 学習によって近似推論を行う有向モデル、生成モデルだ。有向モデルとはデータの流れが決まっているモデルで、データから出力への流れを逆転することが出来ないことを示している。生成モデルとは簡潔に説明すると、入力データのようなデータを生成することが出来るモデルを示している。このモデルを説明するにあたり、まずいくつかの変数を定義する。

- x モデル化したいデータ
- z 潜在変数

- $P(x)$ データの確率分布
- $P(z)$ 潜在変数の確率分布
- $P(x|z)$ 潜在変数を用いて生成されるデータの分布

潜在変数を説明するために一つの例を紹介する。

まず我々は今博物館でモナ・リザを見ているとする。つまりここでモデル化したいデータとしてモナ・リザを設定する。すると我々は脳内のどこかにモナ・リザのイメージを何らかの形式で保存する。この保存されたそれが潜在変数と考えることが出来る。確率分布というのはそれぞれ、博物館の絵画全体、我々が脳内で記憶しているそれらと想像することが出来る。潜在変数を用いて生成されるデータの分布とは、我々が紙を渡されて完璧なモナ・リザを模写する能力を示している。

VAE の目的はデータの分布 $P(x)$ を見つけることにある。これは以下の周辺化の式を用いて計算することが出来る。

$$\begin{aligned} P(x) &= \int P(x,z)dz \\ &= \int P(x|z)P(z)dz \end{aligned} \tag{26}$$

VAE は $P(z|x)$ を用いて $P(z)$ を学習する。

ここで仮定として $p(z)$ を シンプルな (多変量正規) 分布 $N(\mu_z, \sigma_z)$ (例えば $N(0, I)$) とする。

VAE では生成モデルの一種であることから encoder、decoder というものがあり、encoder にデータを通すことにより $P(z)$ を近似した確率分布を得、decoder に確率分布を通すことにより元のデータを生成する。つまり encoder は $P(z|x)$ の近似 $E(z|x)$ 、decoder は $P(x|z)$ の近似 $D(x|z)$ となる。こう仮定することで $p(z|x)$ も多変量正規分布であるという性質が得られる。

さて encoder の学習を考えたとき、$E(z|x)$ の近似の精度を計る尺度として KL ダイバージェンス (Kullback-Leibler divergence) の最小化が登場する。KL ダイバージェンスは近似精度を計る尺度であり、厳密な距離ではないが非負性といった有用な性質がある。

$$\begin{aligned} KL(E(z|x)||P(z|x)) &= \mathbb{E}_{z\sim E(z|x)}[\log \frac{E(z|x)}{P(z|x)}] \\ &= \mathbb{E}_{z\sim E(z|x)}[\log E(z|x) - \log P(x|z) - \log P(z)] + \log P(x) \\ &= KL(E(z|x)||P(z)) - \mathbb{E}_{z\sim E(z|x)}[\log P(x|z)] + \log P(x) \end{aligned} \tag{27}$$

$$\Leftrightarrow$$

$$\log P(x) - KL(E(z|x)||P(z|x)) = \mathbb{E}_{z\sim E(z|x)}[\log P(x|z)] - KL(E(z|x)||P(z)) \tag{28}$$

このとき右辺は ELBO(evidence lower bound) という指標になり、元の KL ダイバージェンスの最小化という問題を ELBO の最大化の問題として考えることが出来る。

更にここで潜在のパラメータベクトル θ を別に導入すると $P(x)$ は以下の式で書き直すことが出来る。(これは VAE モデル全体に共通したパラメータベクトルだ)

$$\begin{aligned} P(x) &= \int P_\theta(x,z)dz \\ &= \int P(x|z;\theta)P(z)dz \end{aligned} \tag{29}$$

このパラメータベクトルは事前に与えられ r るものではないから、学習する $E(z|x)$ は潜在ベクトル ϕ を導入して $E_\phi(z|x)$ となる。

ところでこの学習を行っている際に、encoder である $E_\phi(z|x)$ は z を得るために訓練されることから、$P_\theta(x|z)$ は decoder である $D(x|z;\theta)$ と見なすことが出来る。

さてここから VAE の有名な特徴である、Reparameterization Trick を説明する。

単純に上記の手続きを学習しようとしたとき、多変量正規分布である encoder から z をサンプリングするという手続きが存在しているため、深層学習を用いる際に重要となる誤差逆伝搬を行うことが出来なくなる。これを解決する手法として Reprarameterization Trick がある。z を以下の関数を用いて決定的に生成する。

$$z = \mu_z + \epsilon * \sigma_z \quad where\, \epsilon \sim N(0, I) \tag{30}$$

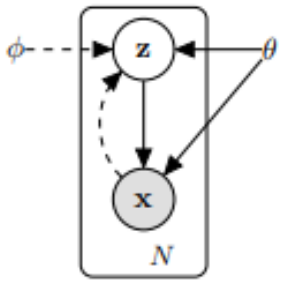

Figure 15: VAE のグラフィカルモデル (Kingma and Welling 2013 より)

9.3 日本語データの取り扱いについて

9.3.1 単語分割

単語分割とは文等を単語単位に分割することであり、例えば英語では一般にスペースで分割することを示す。

日本語で単語分割を行う場合、ビタビアルゴリズム (Viterbi algorithm (Forney 1973)) を使う方法や、双方向 LSTM(Schuster and Paliwal 1997/Graves, Fernández, and Schmidhuber 2005) などの深層学習モデルを用いる手法[21] があるが、前者は新語への対応が難しいこと、後者は前者に比べ処理時間がかかってしまうこと、両者ともに誤る可能性があることが問題となる。

単語分割の例

<英語の場合>
I am a researcher about NLP.
→ [I] [am] [a] [reseacher] [about] [NLP] [.]

<日本語の場合>
私は自然言語処理の研究者です。
→ [私] [は] [自然言語処理] [の] [研究者] [です] [。]

9.3.2 形態素解析

形態素解析とは文に対して形態素と呼ばれる単位に分割、分割したそれぞれに品詞等を割り当てることを指す。本研究や一般的な形態素解析で用いられる形態素という単位は単語であり、ほとんどの場合で形態素解析ができれば単語分割が出来たものとして認められる。この手法としてはルールベースに基づいた手法や、確率的言語モデルに基づいた手法があるが、いずれも完璧ではない。本研究では MeCab を用いたが、こちらは後者を用いた手法で形態素解析を行っている。

[21] https://github.com/taishi-i/nagisa

形態素解析の例

MeCab 0.996 (単語辞書 mecab-ipadic-neologd 20181112-01) で形態素解析を行った。

元の文：日本語の研究をしよう。
解析結果：
日本語 名詞, 一般,*,*,*,*, 日本語, ニホンゴ, ニホンゴ
の 助詞, 連体化,*,*,*,*, の, ノ, ノ
研究 名詞, サ変接続,*,*,*,*, 研究, ケンキュウ, ケンキュー
を 助詞, 格助詞, 一般,*,*,*, を, ヲ, ヲ
しよ 動詞, 自立,*,*, サ変・スル, 未然ウ接続, する, シヨ, シヨ
う 助動詞,*,*,*, 不変化型, 基本形, う, ウ, ウ
。 記号, 句点,*,*,*,*,。,。,。

(順に “表層形 品詞, 品詞細分類 1, 品詞細分類 2, 品詞細分類 3, 活用型, 活用形, 原形, 読み, 発音” となっており、“*” は未定義を示す。)

9.3.3 NER

NER(固有表現抽出 Named Entity Recognition) とは文中から固有表現と分類される固有名詞、数詞、時刻や日付などの情報を抽出し、それぞれに分類するタスクだ。固有表現の例としては、MUC (Message Understanding Conference (Chinchor 1998)) の定義によれば、ORG(組織名)/PER(人名)/LOC(地名)/DATE(日付)/TIME(時間)/MONEY(金額)/PERCENT(割合) の 7 種類がある。この分野は現在も活発に研究が行われており、BERT(Devlin et al. 2018) モデルを用いてファインチューニングでもこのタスクに取り組み発表時点で SOTA を獲得した。

NER の例

元の文：Y さんは 12 月にパソコンを 30 万円で買った。
解析結果： <PER> Y さん </PER> は <DATA> 12 月 </DATE> にパソコンを <MONEY> 30 万円 </MONEY> で買った。

9.3.4 Twitter から収集した対話データの例

以下に Twitter から収集した対話データの例を示す。
個人情報に近いものや出力が困難であるものは以下の記号で置換した。

- [PER]: 人名
- [ORG]: 組織・団体名
- [UNK]: 判別不能なものの恐らく個人情報なもの
- [NUM]: 数字
- [E]: 絵文字・顔文字を含む特殊文字

IN: "[PER] ちゃんが 1 番だけど香子もすき"
OUT: " 唐突な告白かと思ってしまった"

IN:" 唐突な告白かと思ってしまった"
OUT: "[ORG] の [PER] ちゃんです [E] でも [PER] さんもだいすきですよ [E] !! "

IN: "[E] 秋の落ち鮒シーズンだって〜のに…こお天気が [E] クッソ〜 !! [E]"
OUT: "明日は仕事だし、日曜は雨だしやってられないですわ！[E]"

IN: "明日は仕事だし、日曜は雨だしやってられないですわ！[E]"
OUT: "フラストレーションが溜まるいっぽうですね [E]"

IN: "さっきたまにうちの店にくる女の子のお客さんが来たんだけどちっちゃくて可愛いしなんかいい匂いするしお釣り渡すとき手が触れたんだけどめっちゃ柔らかかったしちょっとやばかった"
OUT: "変態じゃん"

IN: "珍しくこの時間に起きてる w"
OUT: "珍しすぎる笑笑"

IN: "人に簡単に死ねとか言う奴いるけど ガチで死んでやろうか?? ほんとに死んだ後に死ねって言ったやつが「え??あいつガチで死んだの??俺のせい？」みたいになってるのを見たい [E] 死んだら見れないけどそうなってるのかとか思ってたら死んでみたいァ”ァ”ァ”ァ”ァ”"
OUT: " おちつけひんぬーはやまるなちびぃ"

IN: "おやすみなさい [E]"
OUT: "おやすみなさいませ [E]"

IN: "[NUM]w[NUM]d 体重 [NUM]kg 前日より+[NUM]kg。スタートから-[NUM]kg。体脂肪率 [UNM]% おはようございます (-_-)zzz ひたすら揚げる作業に入ります… 朝 [NUM] 時から雨マークなんだけど… 運動会やるんかい…"
OUT: " お弁当つくり ファイティン![E]。家族は楽しみにしてるよ!"
———————————————————————

9.3.5 名大会話コーパスのデータ例

以下に名大会話コーパス (逸子, 美恵子, and ディヴィッド義和 2011) のデータ例を示す。

———————————————————————

＠データ１（約３５分）
＠収集年月日：２００１年１０月１６日
＠場所：ファミリーレストラン
＠参加者 F107：女性３０代後半、愛知県幡豆郡出身、愛知県幡豆郡在住
＠参加者 F023：女性４０代後半、岐阜県出身、愛知県幡豆郡在住
＠参加者 M023：男性２０代前半、愛知県西尾市出身、西尾市在住
＠参加者 F128：女性２０代前半、愛知県西尾市出身、西尾市在住
＠参加者の関係：英会話教室の友人
F107：＊＊＊の町というのはちいちゃくって、城壁がこう町全体をぐるっと回ってて、それが城壁の上を歩いても１時間ぐらいですよね。
F023：１時間かからないぐらいだね。
４、５０分で。
F107：そうそう。
ほいでさあ、ずっと歩いていたんだけど、そうすと上から、なんか町の中が見れるじゃん。
あるよね。
ほいでさあ、なんか途中でワンちゃんに会ったんだね。
（ふーん）散歩をしてるワンちゃんに会ったんだ。
F023：城壁の上をやっぱ観光客なんだけどワンちゃん連れてきてる人たち結構多くて。
F107：で、こう、そのワンちゃんと２人を、なに、お父さんとお母さんと歩いて、ワンちゃんに会ったんだ。
途中で。
あワンちゃーんとか言ってなでて、ほいで、この人たちはこっち行って、あたしらこっち行ったじゃん。
ずうーとこうやって回ってきてるの。

また会っちゃって。
ここで。
そうしたら。
F128：おー、そら地球はやっぱり丸かったみたいだね。
F107：そうしたらそのワンちゃんがなんかか喜んじゃって、で、あたしの方に走ってきて、とびついてきちゃってさ。
別にあたしさあ、別にさっきなでただけなのにさあ、なんかすごーいなつかれちゃってね。
F023：さっきね、別に、そんなになでてもいないんだよ。
F107：よしよしって言っただけなのに。
F023：あらワンちゃんだーとか言ってすれ違ったんだよ。
普通に。
それでその次のとき、向こうの方からはーっといってかけてくるじゃん。
F107：すごい勢いで走って。
私、あ、あーさっきの犬だとか私たちが言っとるじゃん。
あんで向こうの人たちも、あっ、さっき会った子たちねみたいな感じで気がついたじゃん。
犬も気がついたじゃん。
じゃははって走ってきちゃって、犬が。
X：＜笑い＞そうなんだ。
＜笑い＞
F107：ほいであちしなんかとびつかれちゃったよ。
X：うそ。
＜笑い＞
F023：＊＊＊って言ってさ。
F107：さっきちょっとなでただけなのにって。
かわいかったね。
...

9.3.6 対話破綻チャレンジの雑談対話コーパスのデータ例

以下に対話破綻チャレンジ (東中, 船越, and 小林 2015) の雑談対話コーパスのデータ例 (json 形式) を示す。
尚一部 "annotations" のデータは紙面の都合上 "[...]" を用いて省略している。
今回特に必要になったタグ名について説明する。

- "annotations" その発話が妥当なものであったかの判定に関する情報を示している。
- "utterance" 実際の発話データを示す。

```
{
    "dialogue-id": "1407219916",
    "group-id": "init100",
    "speaker-id": "12_08",
    "turns": [
        {
            "annotations": [
                {
                    "annotator-id": "01_A",
                    "breakdown": "O",
                    "comment": "",
                    "ungrammatical-sentence": "O"
                },
                {
                    "annotator-id": "01_B",
                    "breakdown": "O",
                    "comment": "",
                    "ungrammatical-sentence": "O"
```

```
        },
        {
            "annotator-id": "02_A",
            "breakdown": "O",
            "comment": "",
            "ungrammatical-sentence": "O"
        },
        {
            "annotator-id": "02_B",
            "breakdown": "O",
            "comment": "",
            "ungrammatical-sentence": "O"
        },
        {
            "annotator-id": "03_A",
            "breakdown": "O",
            "comment": "",
            "ungrammatical-sentence": "O"
        },
        {
            "annotator-id": "04_A",
            "breakdown": "O",
            "comment": "",
            "ungrammatical-sentence": "O"
        },
        {
            "annotator-id": "04_B",
            "breakdown": "O",
            "comment": "",
            "ungrammatical-sentence": "O"
        },
        {
            "annotator-id": "05_A",
            "breakdown": "O",
            "comment": "",
            "ungrammatical-sentence": "O"
        },
        {
            "annotator-id": "05_B",
            "breakdown": "O",
            "comment": "",
            "ungrammatical-sentence": "O"
        },
        {
            "annotator-id": "06_A",
            "breakdown": "O",
            "comment": "",
            "ungrammatical-sentence": "O"
        },
        {
            "annotator-id": "07_A",
            "breakdown": "O",
            "comment": "",
            "ungrammatical-sentence": "O"
        },
        {
```

```
                "annotator-id": "08_A",
                "breakdown": "O",
                "comment": "",
                "ungrammatical-sentence": "O"
            },
            {
                "annotator-id": "08_B",
                "breakdown": "O",
                "comment": "",
                "ungrammatical-sentence": "O"
            },
            {
                "annotator-id": "08_C",
                "breakdown": "O",
                "comment": "",
                "ungrammatical-sentence": "O"
            },
            {
                "annotator-id": "09_A",
                "breakdown": "O",
                "comment": "",
                "ungrammatical-sentence": "O"
            },
            {
                "annotator-id": "09_B",
                "breakdown": "O",
                "comment": "",
                "ungrammatical-sentence": "O"
            },
            {
                "annotator-id": "10_A",
                "breakdown": "O",
                "comment": "",
                "ungrammatical-sentence": "O"
            },
            {
                "annotator-id": "10_B",
                "breakdown": "O",
                "comment": "",
                "ungrammatical-sentence": "O"
            },
            {
                "annotator-id": "11_A",
                "breakdown": "O",
                "comment": "",
                "ungrammatical-sentence": "O"
            },
            {
                "annotator-id": "12_A",
                "breakdown": "O",
                "comment": "",
                "ungrammatical-sentence": "O"
            },
            {
                "annotator-id": "13_A",
                "breakdown": "O",
```

```
                    "comment": "",
                    "ungrammatical-sentence": "O"
                },
                {
                    "annotator-id": "13_B",
                    "breakdown": "O",
                    "comment": "",
                    "ungrammatical-sentence": "O"
                },
                {
                    "annotator-id": "14_A",
                    "breakdown": "O",
                    "comment": "",
                    "ungrammatical-sentence": "O"
                },
                {
                    "annotator-id": "15_A",
                    "breakdown": "O",
                    "comment": "",
                    "ungrammatical-sentence": "O"
                }
            ],
            "speaker": "S",
            "time": "2014-08-05 15:22:54",
            "turn-index": 0,
            "utterance": "こんにちは！声かけてくれるのを待ってたんだ。"
        },
        {
            "annotations": [],
            "speaker": "U",
            "time": "2014-08-05 15:23:07",
            "turn-index": 1,
            "utterance": "今日は最高気温 36 度だって。暑いねえ"
        },
        {
            "annotateions": ["..."],
            "speaker": "S",
            "time": "2014-08-05 15:23:07",
            "turn-index": 2,
            "utterance": "最高気温は１７度が予想されます??"
        },
        {
            "annotations": [],
            "speaker": "U",
            "time": "2014-08-05 15:23:15",
            "turn-index": 3,
            "utterance": "いやいや猛暑ですよ"
        },
        {
            "annotations": ["..."],
            "speaker": "S",
            "time": "2014-08-05 15:23:15",
            "turn-index": 4,
            "utterance": "猛暑に備えてますか？"
        },
        {
```

```
            "annotations": [],
            "speaker": "U",
            "time": "2014-08-05 15:23:32",
            "turn-index": 5,
            "utterance": "うん、日傘さしたり、水飲んだり、クーラー入れたりしてる"
        },
        {
            "annotations": ["..."],
            "speaker": "S",
            "time": "2014-08-05 15:23:32",
            "turn-index": 6,
            "utterance": "クーラーは快適ですね"
        },
        {
            "annotations": [],
            "speaker": "U",
            "time": "2014-08-05 15:23:47",
            "turn-index": 7,
            "utterance": "うん、会社だと朝はつけられないから死にそう"
        },
        {
            "annotations": ["..."],
            "speaker": "S",
            "time": "2014-08-05 15:23:47",
            "turn-index": 8,
            "utterance": "会社は冷房が効きますよねえ"
        },
        {
            "annotations": [],
            "speaker": "U",
            "time": "2014-08-05 15:24:12",
            "turn-index": 9,
            "utterance": "昼間は効くね。ここはパソコンがいっぱいあるからないと大変"
        },
        {
            "annotations": ["..."],
            "speaker": "S",
            "time": "2014-08-05 15:24:12",
            "turn-index": 10,
            "utterance": "動きが悪いです"
        },
        {
            "annotations": [],
            "speaker": "U",
            "time": "2014-08-05 15:24:25",
            "turn-index": 11,
            "utterance": "そうなの？こっちは快適だよ"
        },
        {
            "annotations": ["..."],
            "speaker": "S",
            "time": "2014-08-05 15:24:25",
            "turn-index": 12,
            "utterance": "パソコンは使い勝手が悪いです"
        },
        {
```

```
            "annotations": [],
            "speaker": "U",
            "time": "2014-08-05 15:24:35",
            "turn-index": 13,
            "utterance": "えー、それは使い方がわかってないからだよ"
        },
        {
            "annotations": ["..."],
            "speaker": "S",
            "time": "2014-08-05 15:24:35",
            "turn-index": 14,
            "utterance": "パソコンは調子がいいですね"
        },
        {
            "annotations": [],
            "speaker": "U",
            "time": "2014-08-05 15:24:50",
            "turn-index": 15,
            "utterance": "うん、壊れやすいけどね"
        },
        {
            "annotations": ["..."],
            "speaker": "S",
            "time": "2014-08-05 15:24:50",
            "turn-index": 16,
            "utterance": "パソコンは気分が悪いですか"
        },
        {
            "annotations": [],
            "speaker": "U",
            "time": "2014-08-05 15:25:05",
            "turn-index": 17,
            "utterance": "気分の問題なのかもね"
        },
        {
            "annotations": ["..."],
            "speaker": "S",
            "time": "2014-08-05 15:25:05",
            "turn-index": 18,
            "utterance": "パソコンは元気ですね"
        },
        {
            "annotations": [],
            "speaker": "U",
            "time": "2014-08-05 15:25:13",
            "turn-index": 19,
            "utterance": "うん、私のは元気。"
        },
        {
            "annotations":  ["..."],
            "speaker": "S",
            "time": "2014-08-05 15:25:13",
            "turn-index": 20,
            "utterance": "パソコンは大事ですね"
        }
    ]
```

299 }

9.4 質問文抽出を念頭においた不均衡分散・サイズの分類問題

9.4.1 画像データ

9.4.2 文データ

文データ

9.5 機械翻訳システムを用いた対話

9.5.1 Seq2Seq Attention

9.5.2 Transformer

9.5.3 BLEU スコア

9.6 文のスタイル変換

9.6.1 Sequence to Better Sequence

Sequence to Better Sequence

9.6.2 CopyNet

CopyNet

9.6.3 Denoising Auto Encoder

Denoising Auto Encoder

9.7 CoLA タスクを応用した対話システムのエラー検知

9.7.1 BERT

BERT

10 結論

10.1 今後の課題

今回できなかった文生成の問題・論文に載せることのできなかった推論の内部状態の更新等について言及する。また精度向上や今後取り組みたい問題設定 (Unity などで仮想世界を作り、その中で対話を行えるようにするエージェント作成したい旨) について話す。

References

Bojanowski, Piotr et al. (2017). "Enriching Word Vectors with Subword Information". In: *TACL* 5, pp. 135–146.

Chen, Chun-Yen et al. (2018). *Gunrock: Building A Human-Like Social Bot By Leveraging Large Scale Real User Data*.

Chinchor, Nancy A. (Apr. 1998). "Proceedings of the Seventh Message Understanding Conference (MUC-7) Named Entity Task Definition". In: *Proceedings of the Seventh Message Understanding Conference (MUC-7)*. version 3.5, http://www.itl.nist.gov/iaui/894.02/related_projects/muc/. Fairfax, VA, 21 pages. URL: http://acl.ldc.upenn.edu/muc7/ne_task.html.

Chu, Hang, Daiqing Li, and Sanja Fidler (2018). "A Face-to-Face Neural Conversation Model". In: eprint: arXiv:1812.01525.

Chung, Junyoung et al. (2014). "Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling". In: *CoRR* abs/1412.3555.

Clevert, Djork-Arné, Thomas Unterthiner, and Sepp Hochreiter (2015). "Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs)". In: *CoRR* abs/1511.07289.

Deng, J. et al. (2009). "ImageNet: A Large-Scale Hierarchical Image Database". In: *CVPR09*.

Devlin, Jacob et al. (2018). "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". In: *CoRR* abs/1810.04805.

Elbayad, Maha, Laurent Besacier, and Jakob Verbeek (2018). "Pervasive Attention: 2D Convolutional Neural Networks for Sequence-to-Sequence Prediction". In: *CoNLL*.

Fang, Hao et al. (2018). *Sounding Board: A User-Centric and Content-Driven Social Chatbot*. eprint: arXiv:1804.10202.

Forney, G. D. (Mar. 1973). "The Viterbi algorithm". In: *Proc. of the IEEE* 61, pp. 268–278.

Fukushima, Kunihiko (1980). "Neocognitron: A Self-Organizing Neural Network Model for a Mechanism of Pattern Recognition Unaffected by Shift in Position". In: *Biological Cybernetics* 36, pp. 193–202.

Goodfellow, Ian, Yoshua Bengio, and Aaron Courville (2016a). *Deep Learning*. http://www.deeplearningbook.org. MIT Press. Chap. 10.

— (2016b). *Deep Learning*. http://www.deeplearningbook.org. MIT Press. Chap. 8.

Goodfellow, Ian, Jean Pouget-Abadie, et al. (2014). "Generative Adversarial Nets". In: *Advances in Neural Information Processing Systems 27*. Ed. by Z. Ghahramani et al. Curran Associates, Inc., pp. 2672–2680. URL: http://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf.

Graves, Alex, Santiago Fernández, and Jürgen Schmidhuber (2005). "Bidirectional LSTM Networks for Improved Phoneme Classification and Recognition". In: *Proceedings of the 15th International Conference on Artificial Neural Networks: Formal Models and Their Applications - Volume Part II*. ICANN'05. Warsaw, Poland: Springer-Verlag, pp. 799–804. URL: http://dl.acm.org/citation.cfm?id=1986079.1986220.

Gu, Jiatao et al. (2016). "Incorporating Copying Mechanism in Sequence-to-Sequence Learning". In: *CoRR* abs/1603.06393.

Hu, Zhiting et al. (Aug. 2017). "Toward Controlled Generation of Text". In: *Proceedings of the 34th International Conference on Machine Learning*. Ed. by Doina Precup and Yee Whye Teh. Vol. 70. Proceedings of Machine Learning Research. International Convention Centre, Sydney, Australia: PMLR, pp. 1587–1596. URL: http://proceedings.mlr.press/v70/hu17e.html.

Jain, L. C. and L. R. Medsker (1999). *Recurrent Neural Networks: Design and Applications*. 1st. Boca Raton, FL, USA: CRC Press, Inc. ISBN: 0849371813.

Kalman, Dan (1996). "A singularly valuable decomposition: The SVD of a matrix". In: *College Math Journal* 27, pp. 2–23.

Kingma, Diederik P. and Max Welling (2013). "Auto-Encoding Variational Bayes". In: *CoRR* abs/1312.6114. arXiv: 1312.6114. URL: http://arxiv.org/abs/1312.6114.

Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E Hinton (2012). "ImageNet Classification with Deep Convolutional Neural Networks". In: *Advances in Neural Information Processing Systems 25*. Ed. by F. Pereira et al. Curran Associates, Inc., pp. 1097–1105. URL: http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf.

LeCun, Yann et al. (1999). "Object Recognition with Gradient-Based Learning". In: *Shape, Contour and Grouping in Computer Vision*. London, UK, UK: Springer-Verlag, pp. 319–. ISBN: 3-540-66722-9. URL: http://dl.acm.org/citation.cfm?id=646469.691875.

Luong, Thang, Hieu Pham, and Christopher D. Manning (2015). "Effective Approaches to Attention-based Neural Machine Translation". In: *EMNLP*.

Maaten, Laurens van der and Geoffrey Hinton (2008). “Visualizing Data using t-SNE”. In: *Journal of Machine Learning Research* 9, pp. 2579–2605. URL: http://www.jmlr.org/papers/v9/vandermaaten08a.html.

Mikolov, Tomas, Ilya Sutskever, Kai Chen, Greg S Corrado, et al. (2013). “Distributed Representations of Words and Phrases and their Compositionality”. In: *Advances in Neural Information Processing Systems 26*. Ed. by C. J. C. Burges et al. Curran Associates, Inc., pp. 3111–3119. URL: http://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality.pdf.

Mikolov, Tomas, Ilya Sutskever, Kai Chen, Gregory S. Corrado, et al. (2013). “Distributed Representations of Words and Phrases and their Compositionality”. In: *NIPS*.

Mueller, Jonas, David Gifford, and Tommi Jaakkola (Aug. 2017). “Sequence to Better Sequence: Continuous Revision of Combinatorial Structures”. In: *Proceedings of the 34th International Conference on Machine Learning*. Ed. by Doina Precup and Yee Whye Teh. Vol. 70. Proceedings of Machine Learning Research. International Convention Centre, Sydney, Australia: PMLR, pp. 2536–2544. URL: http://proceedings.mlr.press/v70/mueller17a.html.

Nair, Vinod and Geoffrey E. Hinton (2010). “Rectified Linear Units Improve Restricted Boltzmann Machines”. In: *Proceedings of the 27th International Conference on International Conference on Machine Learning*. ICML’10. Haifa, Israel: Omnipress, pp. 807–814. ISBN: 978-1-60558-907-7. URL: http://dl.acm.org/citation.cfm?id=3104322.3104425.

Pennington, Jeffrey, Richard Socher, and Christopher D. Manning (2014). “Glove: Global vectors for word representation”. In: *In EMNLP*.

Peters, Matthew E. et al. (2018). “Deep contextualized word representations”. In: *NAACL-HLT*.

Sainath, Tara N. et al. (2015). “Convolutional, Long Short-Term Memory, fully connected Deep Neural Networks”. In: *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4580–4584.

Schuster, M. and K.K. Paliwal (Nov. 1997). “Bidirectional Recurrent Neural Networks”. In: *Trans. Sig. Proc.* 45.11, pp. 2673–2681. ISSN: 1053-587X. DOI: 10.1109/78.650093. URL: http://dx.doi.org/10.1109/78.650093.

Serban, Iulian Vlad et al. (2016). *A Hierarchical Latent Variable Encoder-Decoder Model for Generating Dialogues*. eprint: arXiv:1605.06069.

Shen, Tianxiao et al. (2017). “Style Transfer from Non-Parallel Text by Cross-Alignment”. In: *NIPS*.

Sordoni, Alessandro et al. (2015). *A Hierarchical Recurrent Encoder-Decoder For Generative Context-Aware Query Suggestion*. eprint: arXiv:1507.02221.

Sutskever, Ilya, Oriol Vinyals, and Quoc V. Le (2014). “Sequence to Sequence Learning with Neural Networks”. In: *NIPS*.

Vaswani, Ashish et al. (2017). “Attention Is All You Need”. In: *NIPS*.

Vincent, Pascal et al. (2008). “Extracting and Composing Robust Features with Denoising Autoencoders”. In: *Proceedings of the 25th International Conference on Machine Learning*. ICML ’08. Helsinki, Finland: ACM, pp. 1096–1103. ISBN: 978-1-60558-205-4. DOI: 10.1145/1390156.1390294. URL: http://doi.acm.org/10.1145/1390156.1390294.

Warstadt, Alex, Amanpreet Singh, and Samuel R Bowman (2018). “Neural Network Acceptability Judgments”. In: *arXiv preprint arXiv:1805.12471*.

Wo, Xianchao et al. (Mar. 2016). りんな：女子高生人工知能. 言語処理学会 第 22 回年次大会 発表論文集. Microsoft Japan Inc.

東中, 竜一郎, 孝太郎 船越, and 優佳 小林 (Oct. 2015). “対話破綻検出チャレンジ”. In: 言語・音声理解と対話処理研究会 75, pp. 27–32. ISSN: 0918-5682. URL: https://ci.nii.ac.jp/naid/40020632863/.

逸子, 藤村, 大曽 美恵子, and 大島 ディヴィッド義和 (2011). 言語研究の技法：データの収集と分析. Ed. by 藤村逸子、滝沢直宏. ひつじ書房.