

人工知能と自然言語

連続講座「人工知能を科学する」第二回

「機械は考える事が出来るか？」という、最初に掲げた問題が、今では議論にも値しない程無意味なものである事は私も認めよう。

しかし、それにもかかわらず、今世紀の終わりには人達の意見が大きく変化して、ついには、矛盾していると考えることなく『機械の恩考』について語ることができるようになるであろうと私は信じている。

1950年　Turing「機械は考えることができるか？」

しかし六十年代を通しての研究経過を見てみると、計算機は、人間の頭脳労働のある局面を代行することはできるものの、結局、考えることは出来ない機械であり、将来ともに考える機械の実現はほとんど絶望的なことがはっきりしたと恩われるのである。

・・・

さて、常識的な意味で、計算機が”考える機械”でないことははっきりした。これは人類にとって祝福すぺきことであろう。外国語の翻訳をしたり、話しことぱを聞きわけたりする機械の出現もほとんど絶望的である。

1972年　石田晴久「人工知能研究の批判」

1990年代にさしかかると、自然言語処理のコミュニティにおいて、文科系言語学、ひいては記号論的アプローチそのものに対する失望感が蔓延するようになり、その後、統計的アプローチが成功を収めてからは、コミュニティ間の溝は埋めがたいものとなってしまった。

2010年　戸次大介 「文科系言語学と理科系言語学の乖離」

はじめに

- 自由に言語を操る人間の能力を機械で実現することは、人工知能研究にとって、大きな目標の一つである。
  おそらくそれは、人工知能研究の「究極」の目標の一つになるだろうとさえ、僕は考えている。他の動物にはない人間の「知能」の核心部分を構成しているのは、人間の言語能力に他ならないと考えているからである。
- ここ数年で、この分野では大きな前進があった。Googleのニューラル機械翻訳やAlexa等の音声インターフェースを備えたパーソナル・アシスタント・システムの登場は、機械による自然言語処理の歴史の中で、画期をなす出来事といっていい。
- こうした前進を踏まえて、我々は、人間の言語能力の解明に着実に進んでいるのであろうか？とあらためて、問いかけることは意味があるように思う。

はじめに

- コンシューマ・レベルでのプロダクトの普及と人々の経験の広がりは、自然言語処理技術の可能性とともに、その未熟さを、多くの人に気付かせることになるだろうと僕は考えている。
- それは、次の技術の飛躍を準備する重要な契機になる。現在の自然言語処理技術の現状を知ることは、次のステップに進む上で必要なことだ。
- コンピュータで自然言語を扱うアプローチは、多様である。小論では、言語に対する現在の主要なアプローチを、
  1) ディープラーニング技術に基づくもの
  2) 知識表現と検索技術に基づくもの
  3) 計算主義的言語理論に基づくもの

  と、大きく三つのタイプに分けて、それぞれの特徴と動向を紹介しようと思う。

はじめに

- 第一、第二のアプローチについては、これまでのレクチャーで、たびたび述べてきた。小論では、第三の「計算主義的言語理論」に基づくアプローチの紹介に重点を置こうと思う。この第三のアプローチに関心を持っている人は、ITの世界では少ないと感じているからである。
- 僕自身は、文法的に正しい文をリアルタイムに受容し生成する言語能力を「計算能力」として捉える第三のアプローチに魅力を感じている。
- ただ、総体として人間の言語能力を見たとき、「聴く・話す」といった、その「感覚・運動」的側面については、第一のアプローチが、意味や知識を活用する言語能力の「概念・思考」的側面については、第二のアプローチが有効だと考えている。

人工知能と自然言語

Agenda

Part I

ディープラーニングからの自然言語へのアプローチ

Part II

ボイス・アシスタント・システム
Entity Modelと知識検索

Part III

計算主義的言語理論
MinimalistとCategorial Grammar

Part IV

意味の形式的理論

Part I

ディープラーニングからの
自然言語へのアプローチ



Part I
ディープラーニングからの自然言語へのアプローチ　概説

- ディープラーニングからの自然言語へのアプローチでの大きな成功例は、Google ニューラル機械翻訳と、Hintonらが切り開いたディープラーニングを用いた音声認識技術である。後者については、ボイス・アシスタントの基礎技術であり、Part II の冒頭で取り上げる。この Part Iでは、主要に、ニューラル機械翻訳技術の進化をあとづける。
- ニューラル機械翻訳の最初のアイデアは、2004年のBengioの論文で提示される。それは、それ以前の機械翻訳の主流であった「統計的機械翻訳モデル」への批判的考察に基づいたものであった。ただ、そのアイデアがニューラル機械翻訳として実を結ぶまで、10年以上の時間が必要だった。

Part I
ディープラーニングからの自然言語へのアプローチ　概説

- ニューラル機械翻訳のアーキテクチャーは、基本的に、Encoder-Decoder Modelに基づいている。Googleのニューラル機械翻訳は、先行したいくつかの研究を忠実に受け継いだものである。
- DNC(Differentiable Neural Computer)は、小学生程度の国語の問題を機械に解かせようという試みである。一定の成功を収める。これらの訓練に利用されたbabIデータセットの日本語版の作成は、日本でのこうした研究・開発にとって不可欠であると僕は考えている。
- DNCは、アーキテクチャー的には、ディープラーニングのエンジンに、外部メモリー（行列）を持たせようとする。ただし、そのメモリー・アクセスを含めて、徹頭徹尾、ディープラーニングの手法にこだわる。

Part I Agenda

ディープラーニングからの自然言語へのアプローチ

- ニューラル確率言語モデル -- Bengioの「次元の呪い」
- Encoder / Decoder -- HintonのAutoencoder
- Word2Vec -- Mikolov 語の「意味ベクトル」
- Sequence to Sequence -- Ilya Sutskever
- Attention Mechanism -- Bahdanau
- Google ニューラル機械翻訳 -- Yonghui Wu
  - WordPiece
- Google 多言語ニューラル機械翻訳 -- Melvin Johnson
- Differentiable Neural Computer -- Alex Graves
  - bAbI task

ニューラル確率言語モデル
Bengioの「次元の呪い」

従来の機械翻訳の主流であった「統計的機械翻訳モデル」に代わる、「ニューラル機械翻訳モデル」を提案したのは、次のBengioの論文である。

2003年　Yoshua Bengio et al.

“A Neural Probabilistic Language Model”

http://goo.gl/977AQp

Benjioの「次元の呪い」

- Bengioは、早くから、統計的機械翻訳モデルに現れる組み合わせの数の爆発を意識していた一人である。彼は、それを「次元の呪い」 Curse of Dimentionality と呼んだ。
- 26文字のアルファベット15文字以内で構成される語の数は、高々、26^15である。スペル訂正の次元は、その程度のオーダーである。ただし、語彙が10万個ある言語での10個の語からなる文は、100000^10＝10^50種類もある！
- この$_1$ 文$_2$ は$_3$ 10$_4$ 個$_5$ の$_6$ 語$_7$ から$_8$ できて$_9$ いる$_{10}$
  のように、10語文というのは、そんなに長い文章ではない。が、10^50というのは、とてつもなく巨大な数である。
- ただ、こうした問題を、「呪い」としてではなく、「言語の創造性」として、解決すべき「課題」として、当初から理論構成して来たのは、Chomskyらである。

2003年 Yoshua Bengio et al. “A Neural Probabilistic Language Model” http://goo.gl/977AQp

- 「統計的言語モデルの目標は、ある言語における語のシーケンスの結合確率関数を学習することである。ただ、これは、次元の呪いのために本質的に困難である。その上でモデルが試される語のシーケンスは、モデルが学習中に出会った全ての語のシーケンスとは異なっている可能性が高い。」
- 「伝統的だが非常に成功したn-gramベースのアプローチは、訓練データ中に現れた非常に短いシーケンスの重なりを連結することで、一般化の能力を得ている。それぞれの訓練用データ中の文が、モデルに、指数関数的な数のその文と意味的に近い文の情報を伝えることを可能にするような、語の分散表現を学習することで、この次元の呪いと戦うことを提案する。」

- 具体的には、この論文で、彼は、次のような方法を提案する。

1. 語彙中のそれぞれの語に、Rmに実数値の値を持つ、分散した語の特徴ベクトル(word feature vector)を対応づける。
2. 語の並びの結合確率関数を、この並びの中の語の特徴ベクトルで表現する。
3. 語の特徴ベクトルとこの確率関数のパラメーターを、同時に学習する。

- 要は、統計的モデルのように、単に語の並びの統計的性質に依拠するだけでなく、それぞれの語に「特徴ベクトル」という語の「意味」の対応物を導入しようということだと僕は理解している。

語と文の複雑さの違いについて

- 統計的に言語にアプローチする際に、重要なことは、Benjioが指摘したように、語と文とでは、複雑さの次元が全く異なると言うことである。
- 先には、語彙の数を10万として計算したが、現実には語彙の数は、もっと多い。日本語の辞書の収録語彙数をあげておく。
  - 『日本国語大辞典』(小学館)　50万語
  - 『広辞苑』(第六版、岩波書店)　約24万語
  - 『岩波国語辞典』(第七版)　6万5000語
- それでも、語の数は有限である。それに対して、文の数は、可能的には無限である。すべての文を網羅した用例集は、存在しえない。ただし、可能な語のすべての組み合わせを考える必要はない。「文法」が、その構造を与えている。

Encoder / Decoder
HintonのAutoencoder

ニューラル・ネットワークの中で、Encoder-Decoderの枠組みを提案した基本的な論文は、Hintonの 2006年の **”Reducing the Dimensionality of Data with Neural Networks”** である。https://goo.gl/Ki41L8

Googleのニューラル機械翻訳は、この枠組みを利用している。

2006年 Hinton ”**Reducing the Dimensionality of Data with Neural Networks**” https://goo.gl/Ki41L8

「高次元の入力ベクターを再構成する小さな中央層を持つ多層のニューラル・ネットワークを訓練することで、高次元のデータを低次元のコードに変換することができる。

このようなAuto Encoderネットワークの重みを調整するのに勾配降下法を利用できる。ただし、それは、重みの初期値が、良好な解に近い場合にのみうまく働く。

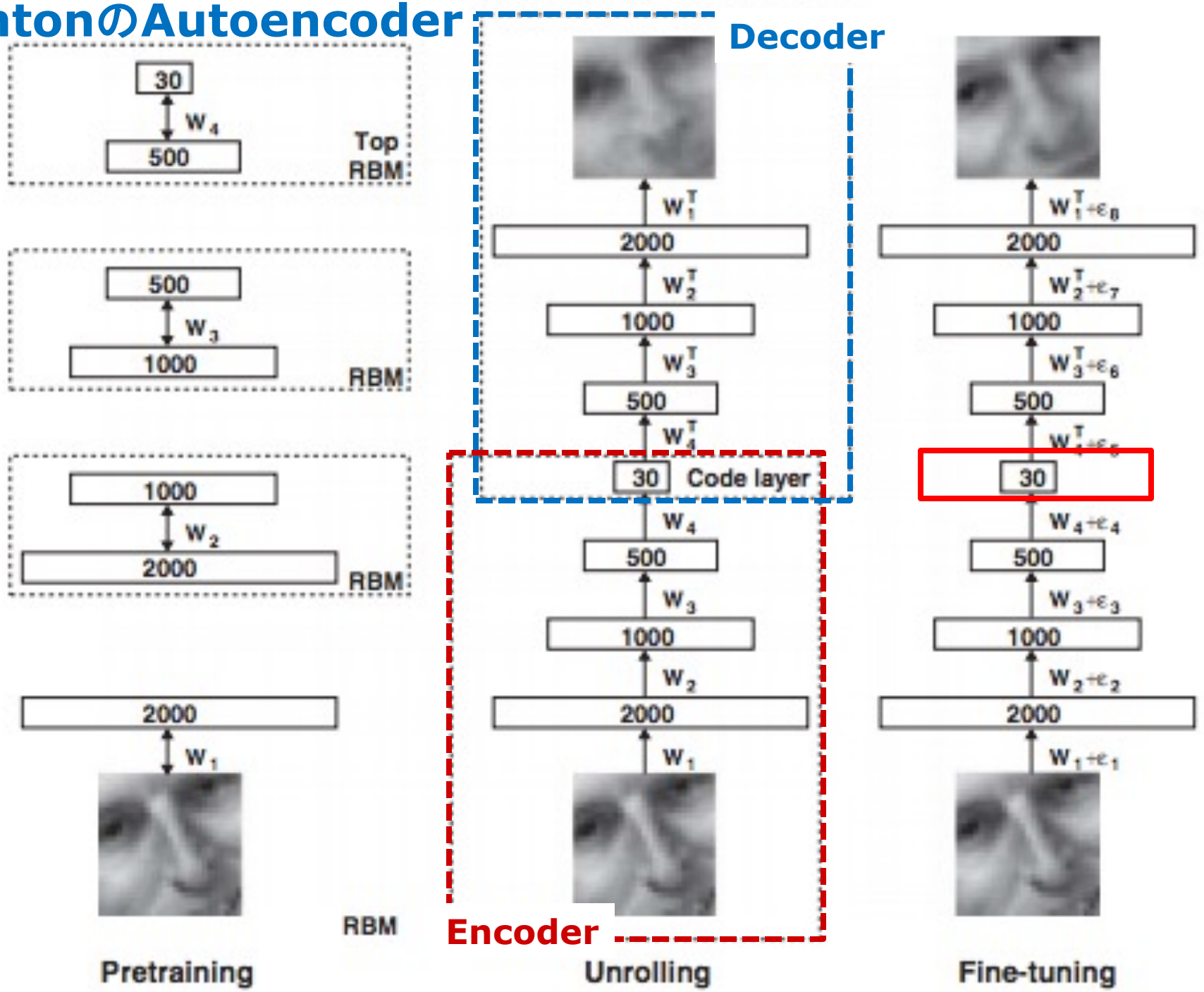
我々は、ディープAuto Encoderネットワークが、低次元のコードを学習することを可能にする重みの初期化の効率的な方法について述べる。この方法は、データの次元を下げるツールとしての主成分分析より、ずっと優れている。」

Autoencoder

- 次の図の中央、下の方の赤い枠がEncoderである。Encoderは、2000次元のベクトル（2000 pixelの画像データ）を、30次元のベクトルに変える。上の方の青い枠のDecoderは、この30次元のベクトルから、2000次元のベクトルを生成する。（こうして、画像が復元される）論文では、このAutoencoderを微調整する方法が示されているのだが（下図の右側。画質が改善されている）、それについては割愛する。

- 注目して欲しいのは、ここでは、入力に与えられたデータ自身が、教師用のデータの役割を果たすので、その意味では、ラベルづけられた教師用のデータを必要としないということ。Autoencoderとは、「**自己エンコーダ**」の意味である。

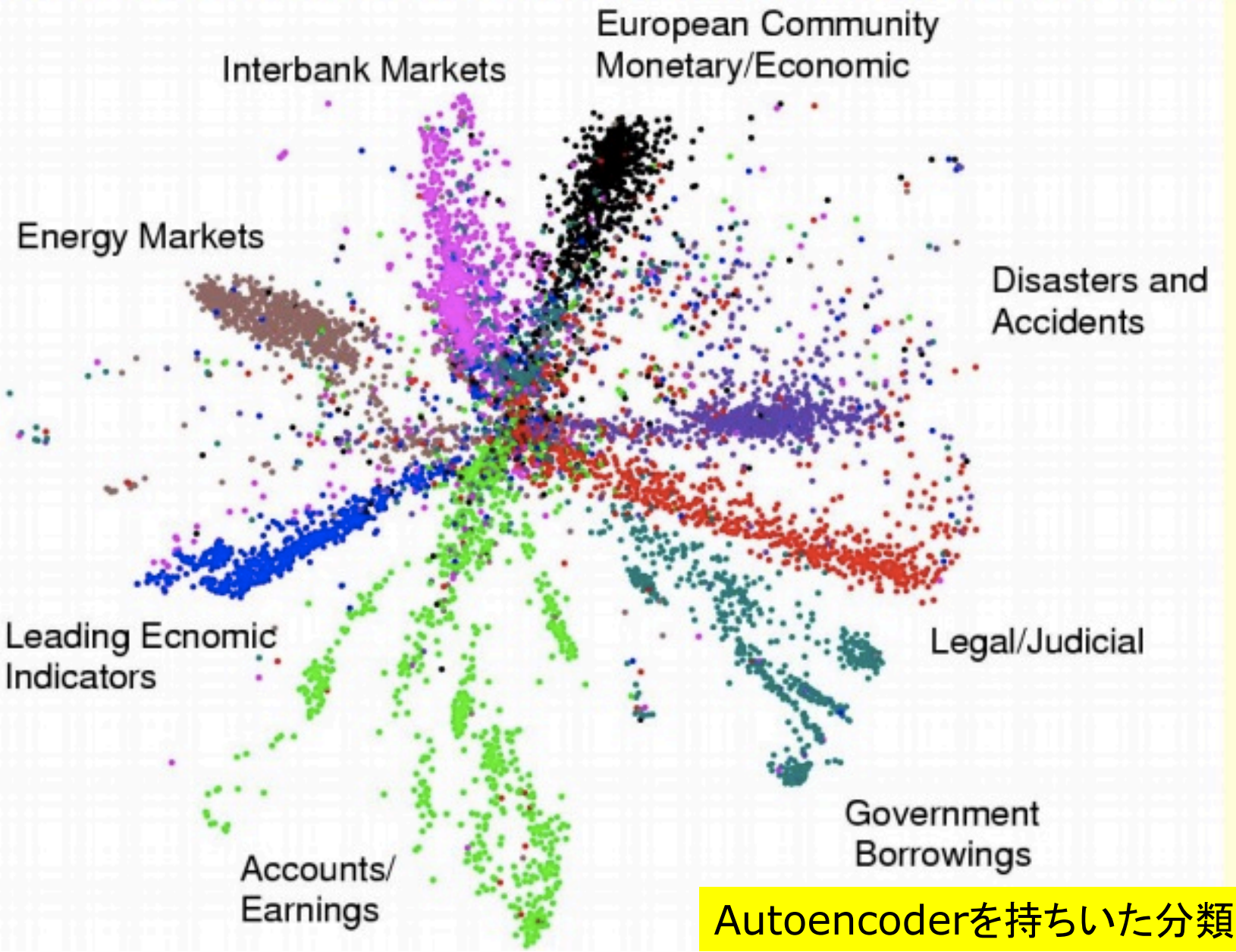
HintonのAutoencoder



書籍の分類へのAutoencoderの利用

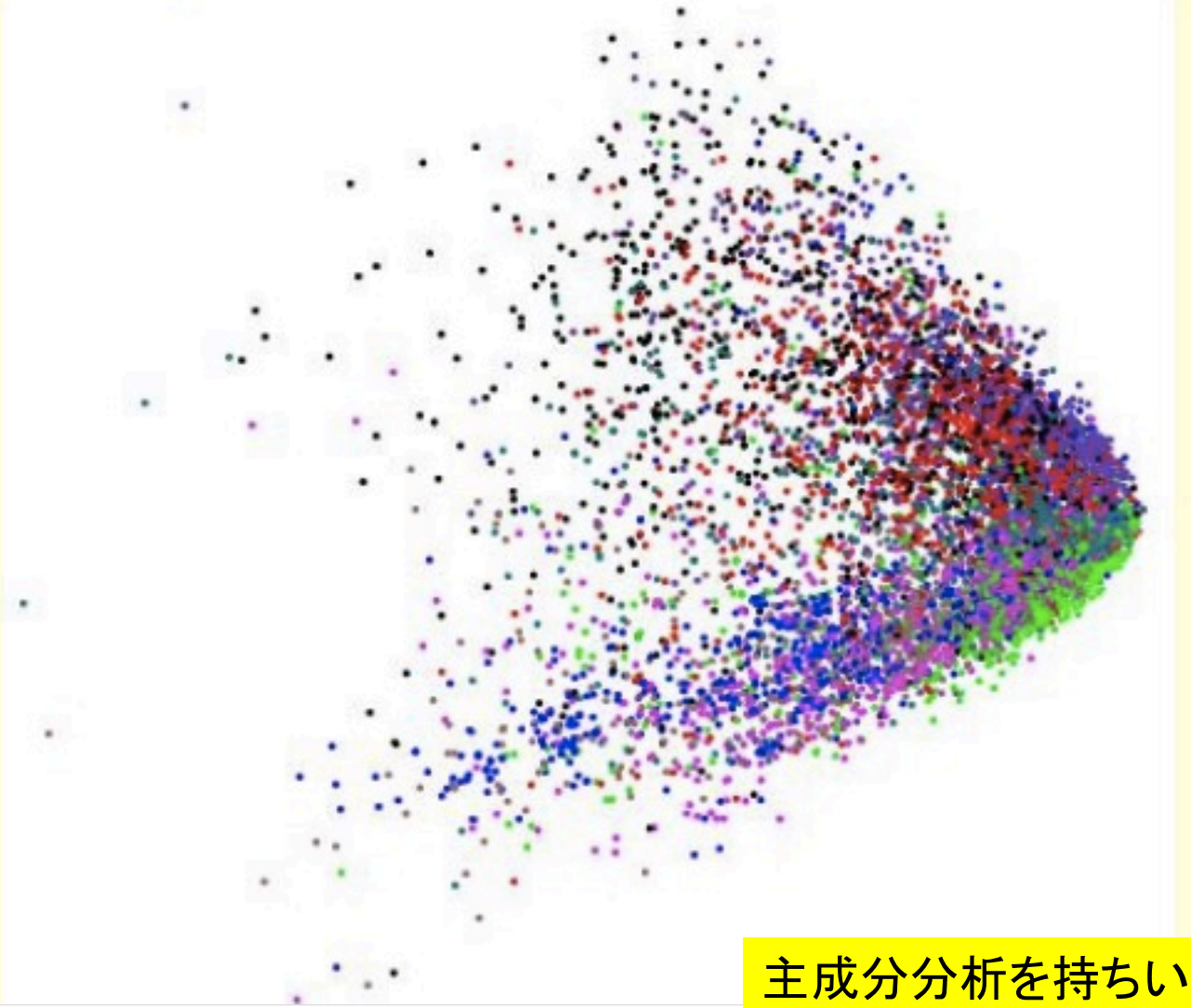
- 先の例は、「画像圧縮技術」の一種として理解してもいいのだが、この論文で、Hintonは、もっと面白い例を紹介している。書籍の分類に、このAutoencoderを使おうというものである。
- よく使われる単語を2000個ほど選ぶ。ある本にこれらの単語が何個含まれているかをカウントする。そうすると、ある本に2000次元の整数からなるベクトルを対応づけることができる。このベクトルをAutoencoderの入力に与えて、Autoencoderがこのベクトルを出力に再現できるように訓練をする。Autoencoderの中央のボトルネックの部分を、10次元のベクトルにすると、ある本に10個の数字を対応づけることができる。
- Hintonは、40万冊のビジネス書を対象に、この方法で得られた10個の数字が、書籍の分類に有効かどうかを実験した。結果を、二次元に可視化したものが、次の図だ。見事に、分類に成功している。

First compress all documents to 2 numbers.
Then use different colors for different document categories



Autoencoderを持ちいた分類

First compress all documents to 2 numbers using a type of PCA
Then use different colors for different document categories



主成分分析を持ちいた分類

Semantic hashing（意味的ハッシング）

- 重要なことは、「画像」と「書籍」では、対象のデータの性質はまるで異なるのだが、Autoencoderは、そのいずれに対しても、高次元のデータを低次元のデータに変換しているということである。別の言葉で言えば、**それは、対象の高次元のデータから、低次元のデータを、元の情報のエッセンスとして取り出しているのである。**
- Hintonは、こうしたAutoencoderの働きを、**Semantic hashing（意味的ハッシング）**と呼んでいる。SHA-1のようなハッシングでは、ハッシュ化されたデータから元のデータを復元することは不可能なのだが、Semantic hashingされたデータは、データの次元は低いものの、元の情報の中核部分を保持している。
- のちに、機械翻訳で利用される所以である。

RNNによる文の生成

"Generating Text with Recurrent Neural Networks"

Ilya Sutskever et al.

http://goo.gl/vHRHSn

2011年



RNNの驚くべき能力

"The Unreasonable Effectiveness of Recurrent Neural Networks"

Andrej Karpathy

http://goo.gl/mNqwCv

2015年



Word2Vec -- 語の「意味ベクトル」

“Linguistic Regularities in Continuous Space Word

Tomas Mikolov et al.

http://goo.gl/j25y8s

2013年 Tomas Mikolov et al.
“Linguistic Regularities in Continuous Space Word http://goo.gl/j25y8s

「連続スペース言語モデルは、最近、さまざまなタスクにわたって優れた結果を示している。本論文では、入力層の重みによって暗黙的に学習されたベクトル空間での単語表現を調べる。

これらの表現は、言語の文法規則と意味規則を把握する上で驚くほど優れており、それぞれの関係はその関係に特有のベクトルオフセットによって特徴付けられることがわかる。

これにより、単語間のオフセットに基づいて、ベクトル指向の推論が可能になる。例えば、男性/女性の関係性は自動的に学習され、そこから導かれたベクトル表現では、”KING – MAN + WOMAN” は、”QUEEN” に非常に近いベクトルになる。」

どんな語が、与えられた語の近くに埋め込まれるか？

FRANCE	JESUS	XBOX	REDDISH	SCRATCHED	MEGABITS
AUSTRIA	GOD	AMIGA	GREENISH	NAILED	OCTETS
BELGIUM	SATI	PLAYSTATION	BLUISH	SMASHED	MB/S
GERMANY	CHRIST	MSX	PINKISH	PUNCHED	BIT/S
ITALY	SATAN	IPOD	PURPLISH	POPPED	BAUD
GREECE	KALI	SEGA	BROWNISH	CRIMPED	CARATS
SWEDEN	INDRA	PSNUMBER	GREYISH	SCRAPED	KBIT/S
NORWAY	VISHNU	HD	GRAYISH	SCREWED	MEGAHERTZ
EUROPE	ANANDA	DREAMCAST	WHITISH	SECTIONED	MEGAPIXELS
HUNGARY	PARVATI	GEFORCE	SILVERY	SLASHED	GBIT/S
SWITZERLAND	GRACE	CAPCOM	YELLOWISH	RIPPED	AMPERES

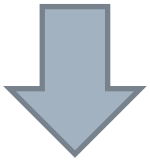
Collobert *et al.* (2011) http://arxiv.org/pdf/1103.0398v1.pdf

どんな語が、与えられた語の近くに埋め込まれるか？

- 似た意味を持つ言葉は、似たベクトルを持つ。
- 似た言葉で置き換えても、正しい文は、正しい文に変わる。

“a **few** people sing well”　　正しい文





“a ***couple*** people sing well”　　正しい文

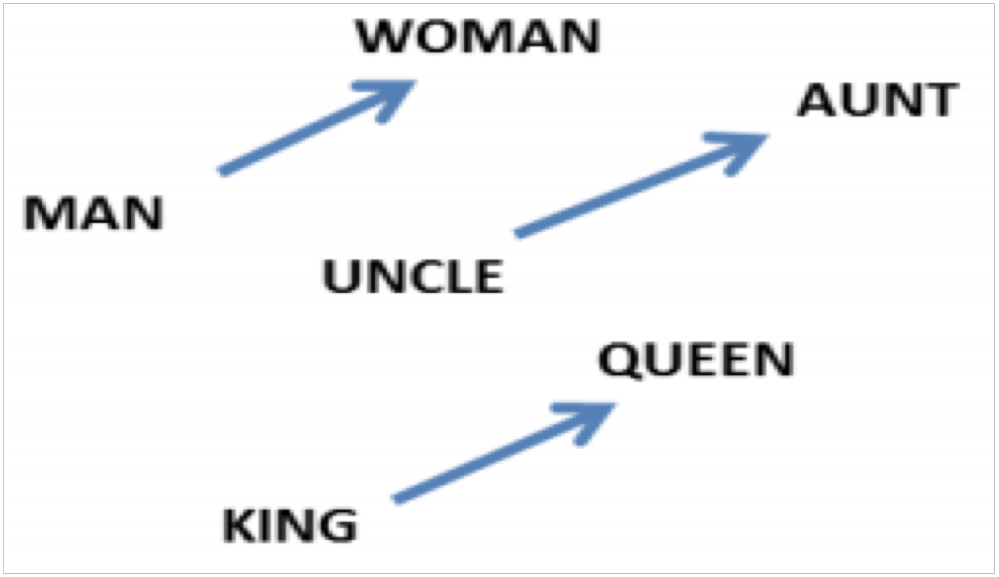
- 意味が似ていなくても、同じクラスの言葉で置き換えても、正しい文は、正しい文に変わる。

“the **wall** is **blue**”  “the ***ceiling*** is ***red***”

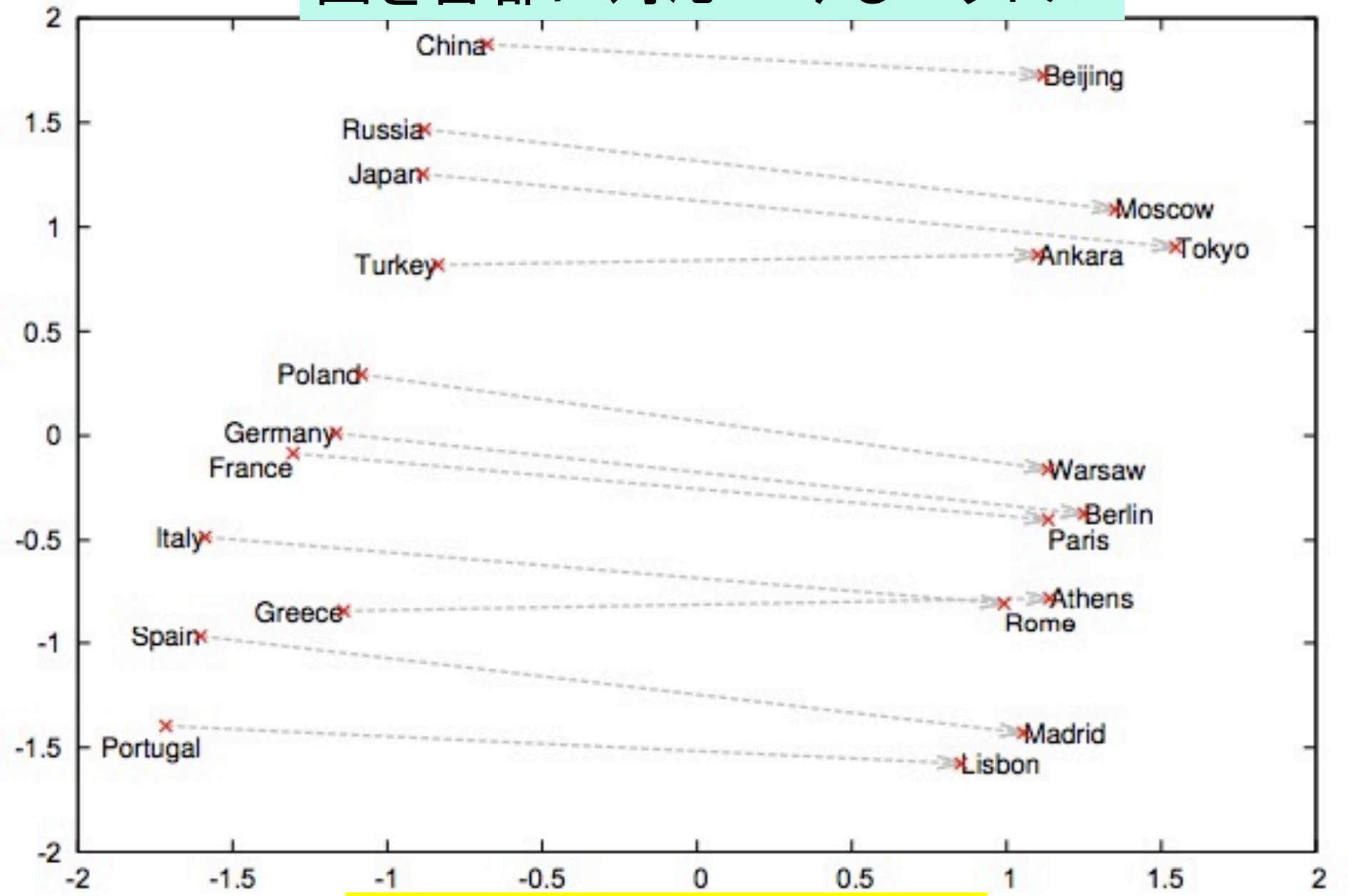
意味を変換するベクトルは共通？

- Word Embeddingは、もっと面白い性質を持つ。下の図のように、男性から女性へのベクトルがあるように見える。

- W(“woman“)−W(“man”) ≃ W(“aunt“)−W(“uncle”)
  W("woman")−W("man") ≃ W("queen")−W("king")



国を首都に対応づけるベクトル



http://arxiv.org/pdf/1310.4546.pdf

ベクトルで表現される関係の例

Relationship	Example 1	Example 2	Example 3
France - Paris	Italy: Rome	Japan: Tokyo	Florida: Tallahassee
big - bigger	small: larger	cold: colder	quick: quicker
Miami - Florida	Baltimore: Maryland	Dallas: Texas	Kona: Hawaii
Einstein - scientist	Messi: midfielder	Mozart: violinist	Picasso: painter
Sarkozy - France	Berlusconi: Italy	Merkel: Germany	Koizumi: Japan
copper - Cu	zinc: Zn	gold: Au	uranium: plutonium
Berlusconi - Silvio	Sarkozy: Nicolas	Putin: Medvedev	Obama: Barack
Microsoft - Windows	Google: Android	IBM: Linux	Apple: iPhone
Microsoft - Ballmer	Google: Yahoo	IBM: McNealy	Apple: Jobs
Japan - sushi	Germany: bratwurst	France: tapas	USA: pizza

Mikolov et al (2013) http://arxiv.org/pdf/1301.3781.pdf

Encoder-Decoderの機械翻訳への応用

HintonのAutoencoderは、そのままの形では、機械翻訳に利用されることはなかったと思う。ただ、2014年に、Ilya Sutskever らは、RNN（ここでは多層のLSTM）の持つシーケンスをシーケンスに変換する能力が、機械翻訳に応用できるという論文 **"Sequence to Sequence Learning with Neural Networks"** https://goo.gl/U3KtxJ を発表する。

2014年 Sutskever et al. "**Sequence to Sequence Learning with Neural Networks"** https://goo.gl/U3KtxJ

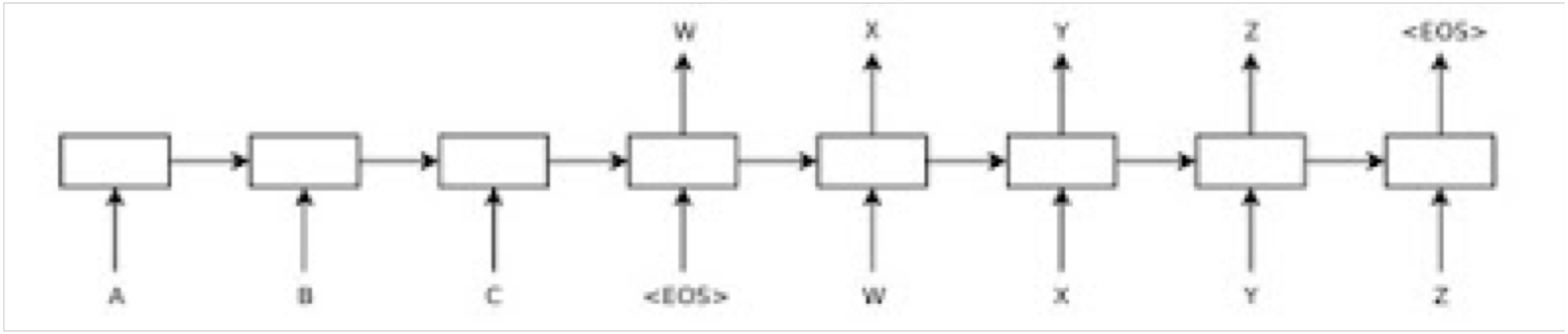
「ディープニューラルネットワーク（DNN）は、難しい学習課題でも優れたパフォーマンスを達成する強力なモデルである。DNNは、ラベル付けられた訓練用のデータが利用可能な時には、いつもうまく機能するというものの、シーケンスをシーケンスにマップすることに、DNNを利用することはできない。

我々は、この論文で、シーケンスを学習するエンド・ツゥ・エンドの汎用のアプローチを提示する。そこでは、シーケンスの構造に最小限の前提しか課していない。

我々の方法では、入力のシーケンスを固定次元のベクトルにマップするのに、多層のLong Short-Term Memory（LSTM）を利用する。その後、別の深いLSTMが、このベクトルから目的のシーケンスをデコードする。」

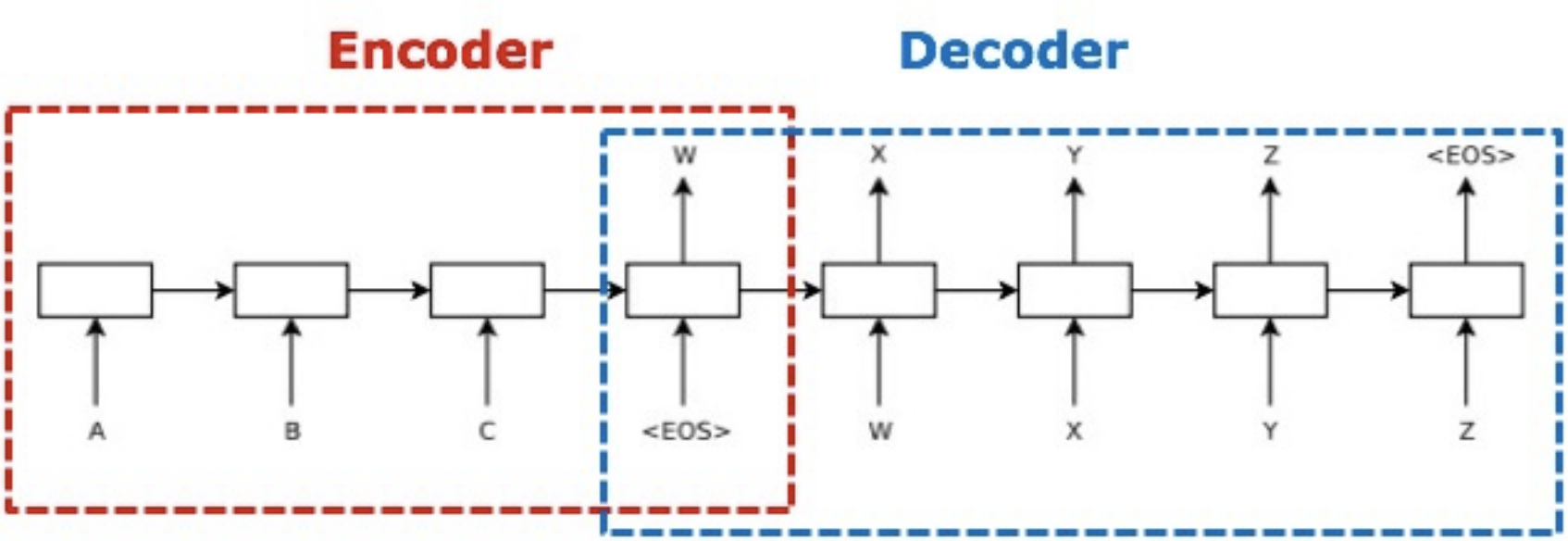
Sequence to Sequence

- この論文の次の図を見て欲しい。

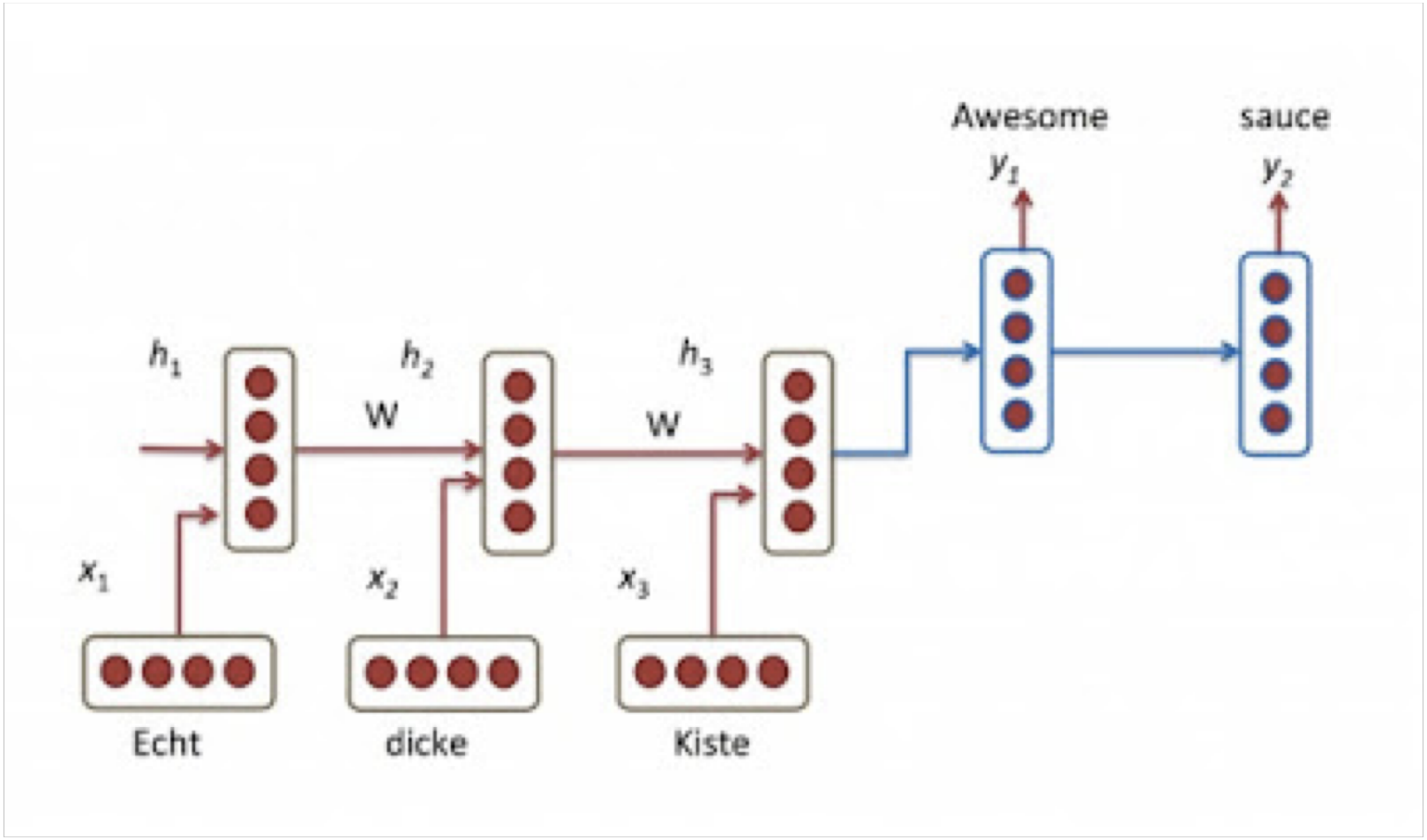


- この図は、このシステムが、ABCというシーケンスが与えられた時、xyzというシーケンスを返すことを表している。<EOS>は、End of Sequence でシーケンスの終わりを表す特別な記号である。（これが、シーケンスの構造に課せられた「最小限の前提」である。）

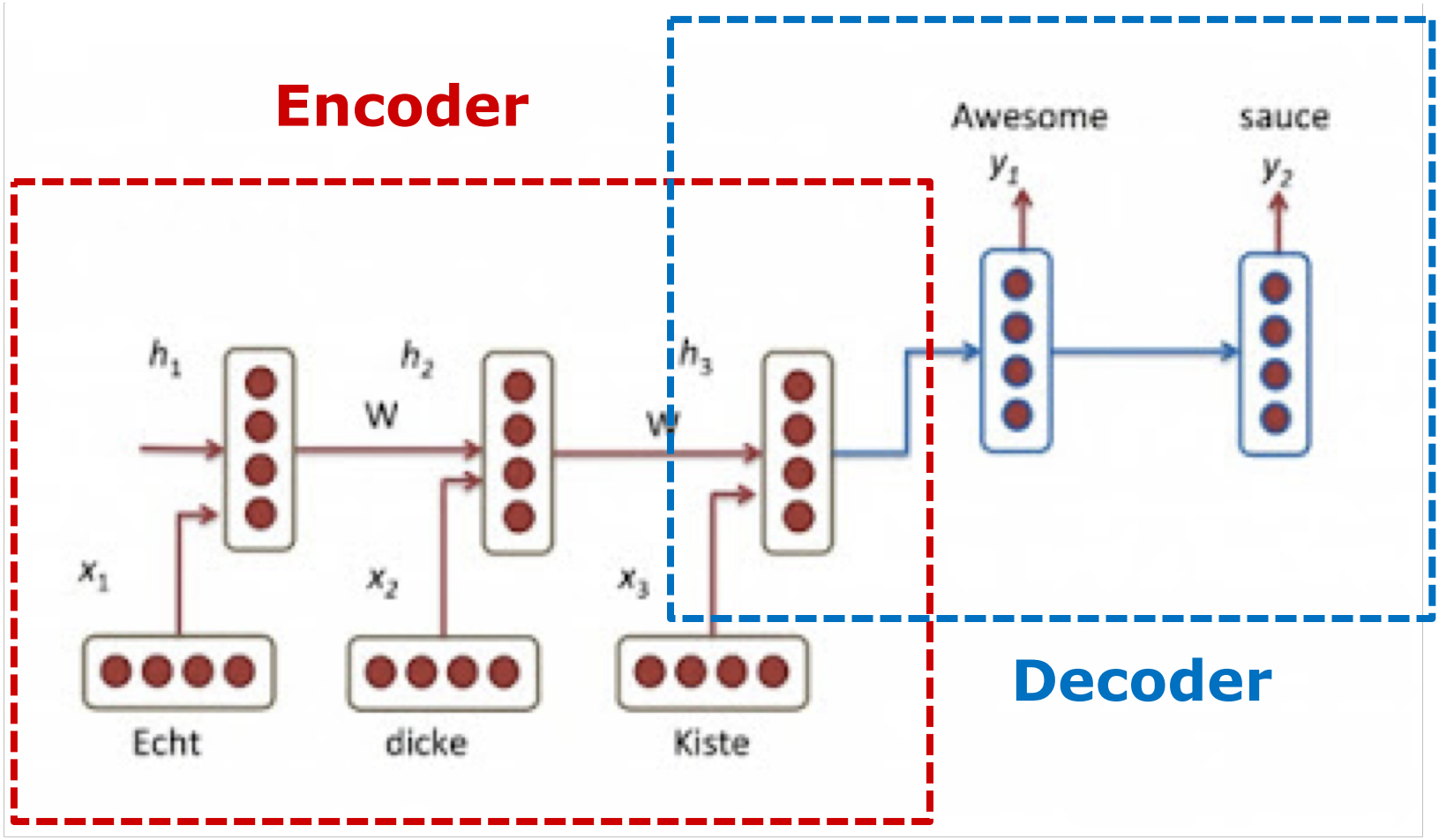
- これが、先に見たEncoder-Decorderのパターンであることは、次のようにしてわかる。
- 先行するLSTM群は、入力シーケンスABCを受け取って、それを固定長のベクトwに変換している。後行のLSTM群は、そのベクトルwを受け取って、それから出力シーケンスxyzを生成する。すなわち、先行のLSTM群をEncoder、後行のLSTM群をDecoderと考えることができる。中間に生成され、両者で共有されるwは、先のHintonのボトルネック部だと思えばいい。



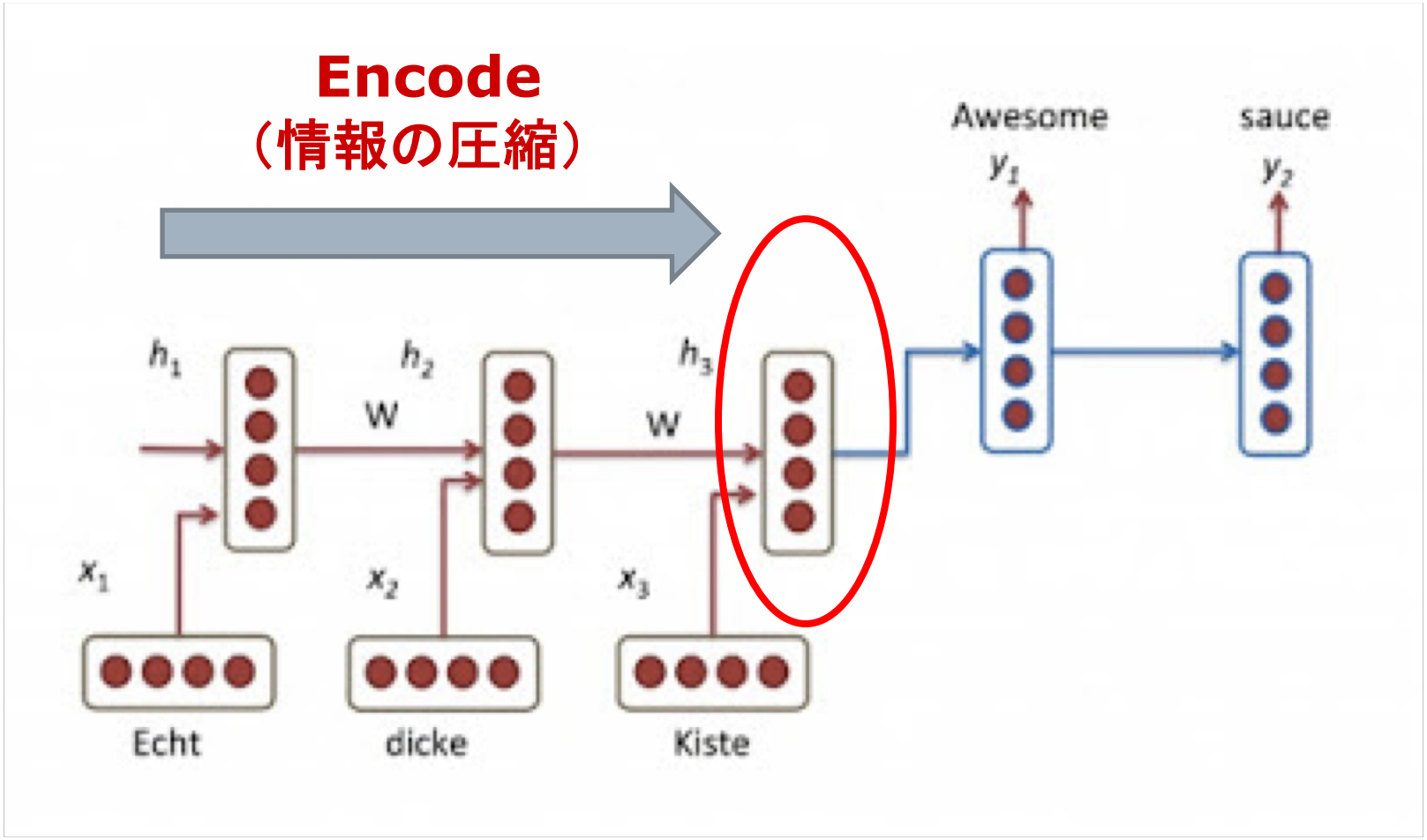
- 次の図(https://goo.gl/JGckBP から)は、こうしたメカニズムで、RNNが、独文の "Echt dicke Kiste" を英文の "Awesome sauce" に翻訳する様子を表している。(ここでは、文章の終わりを表す<EOS>は、省略されている)



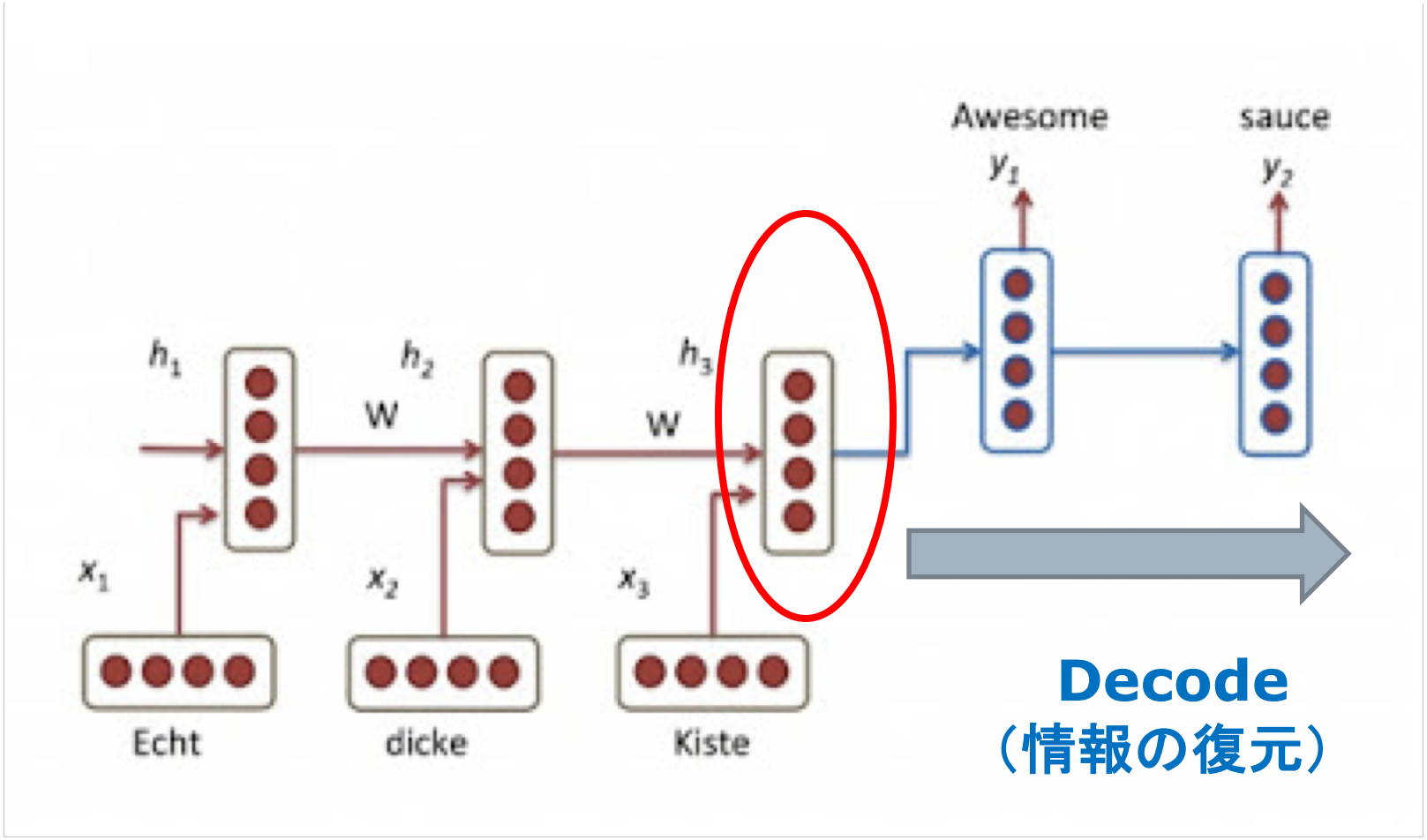
- 次の図(https://goo.gl/JGckBP から)は、こうしたメカニズムで、RNNが、独文の "Echt dicke Kiste" を英文の "Awesome sauce" に翻訳する様子を表している。(ここでは、文章の終わりを表す<EOS>は、省略されている)



- ここでは、Encoder部が、文章の最後にとるRNNの内部状態h_3が、そのままDecoder部に渡されることが示されている。入力シーケンスの情報のエッセンスが、この内部状態 h_3に凝縮されていると考えればいい。



- AutoencoderのDecoder部が、圧縮された情報から元の情報を復元しようとするように、ここでは、その情報から、**「同じ意味」**を持つ、別の言語の文章を復元しようとする。



- Ilya Sutskever らは、このアーキテクチャーで、英語をフランス語に翻訳するシステムを作成し、BLEUのスコアで、34.81という高得点をたたき出した。

- この時のシステムは、5段重ねのLSTMで構成され、それぞれが 8,000次元の状態からなる384M個のパラメーターを持つものだった。

Attention Mechanism

新しいGoogle翻訳のシステムは、全く新しいアイデアに基づいてスクラッチから作り上げられたものではない。その基本的なアイデアは、2016年の5月にarXivに投稿された次の論文に多くを負っている。それは、先の、Ilya Sutskever らのシステムを発展させたものだ。

Bahdanau, D., Cho, K., and Bengio, Y. **“Neural machine translation by jointly learning to align and translate”** https://goo.gl/HZxbNH

Bahdanau, D., Cho, K., and Bengio, Y. **“Neural machine translation by jointly learning to align and translate”** https://goo.gl/HZxbNH

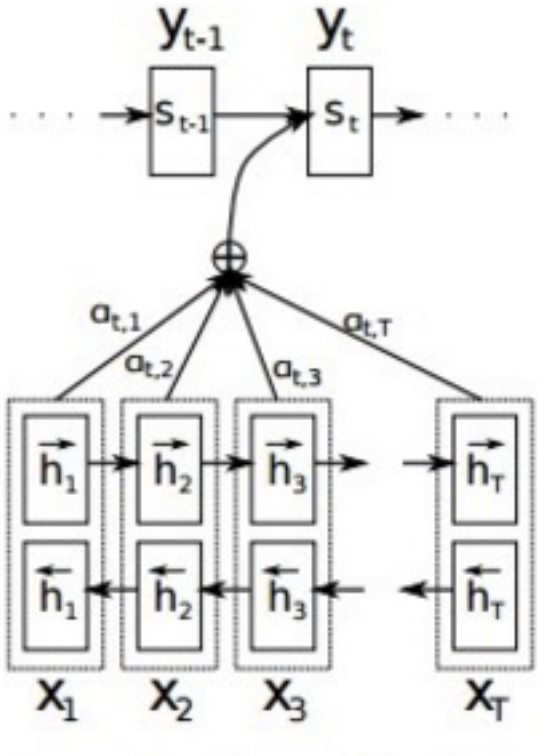
「近年、ニューラル機械翻訳として提案されたモデルは、多くの場合、Encoder-Decoderのファミリーに属している。そこでは、ソースの文が固定長ベクトルにエンコードされ、そこからデコーダが 翻訳文を生成する。この論文では、固定長ベクトルの使用が、この基本的なEncoder/Decoderアーキテクチャの性能を改善する上でのボトルネックになっていると推論し、モデルに自動的に、ターゲット・ワードを予測するのに重要なソース・文の一部分について、（ソフト）検索を可能とすることによって、これを拡張することを提案する。その際、これらの部分を明示的にハードセグメントとして形成する必要はない。」

固定長ベクトルがボトルネック

- 先に見た、Ilya Sutskever らの翻訳システムでは、翻訳されるべき文は、Encoderで、一旦、ある決まった大きさの次元（例えば8000次元）を持つベクトルに変換される。このベクトルからDecoderが翻訳文を生成する。入力された文が、長いものであっても短いものであっても、中間で生成され以降の翻訳プロセスすべての出発点となるこのベクトルの大きさは同じままだ。このシステムでは、長くても短くても入力された文全体が、一つの固定長のベクトルに変換されるのだ。
- 確かに、そこは翻訳の精度を上げる上でのボトルネックになりうる。事実、Ilya Sutskever らのシステムでは、文の長さが長くなるにつれて、翻訳の精度が低下されるのが観察されるという。

この論文の基本的アイデア

- 文全体に一つの固定長のベクトルを割り当てるのではなく、Decoderが翻訳時に、ソース・文の一部分を 改めて見直して、その部分から提供される情報を翻訳に生かそうということだ。



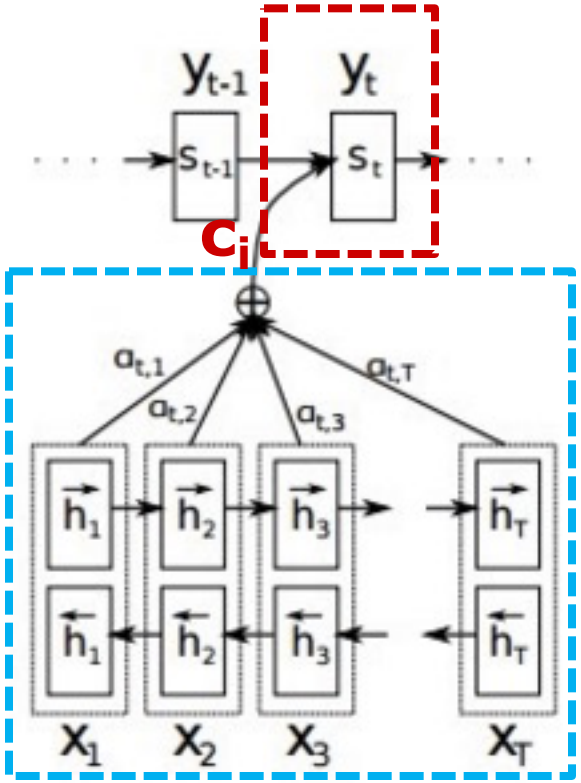
$a_{3,2}$が大きい場合、これは、Decorderがターゲット文の第3の単語を生成しながら、ソース文の第2の状態に多くの注意を払うことを意味する。

「ここで、yはデコーダによって生成された翻訳された単語であり、xは原文の単語である。上記の図は双方向のリカレント・ネットワークを使用しているが、それは重要ではない。逆方向は無視していい。

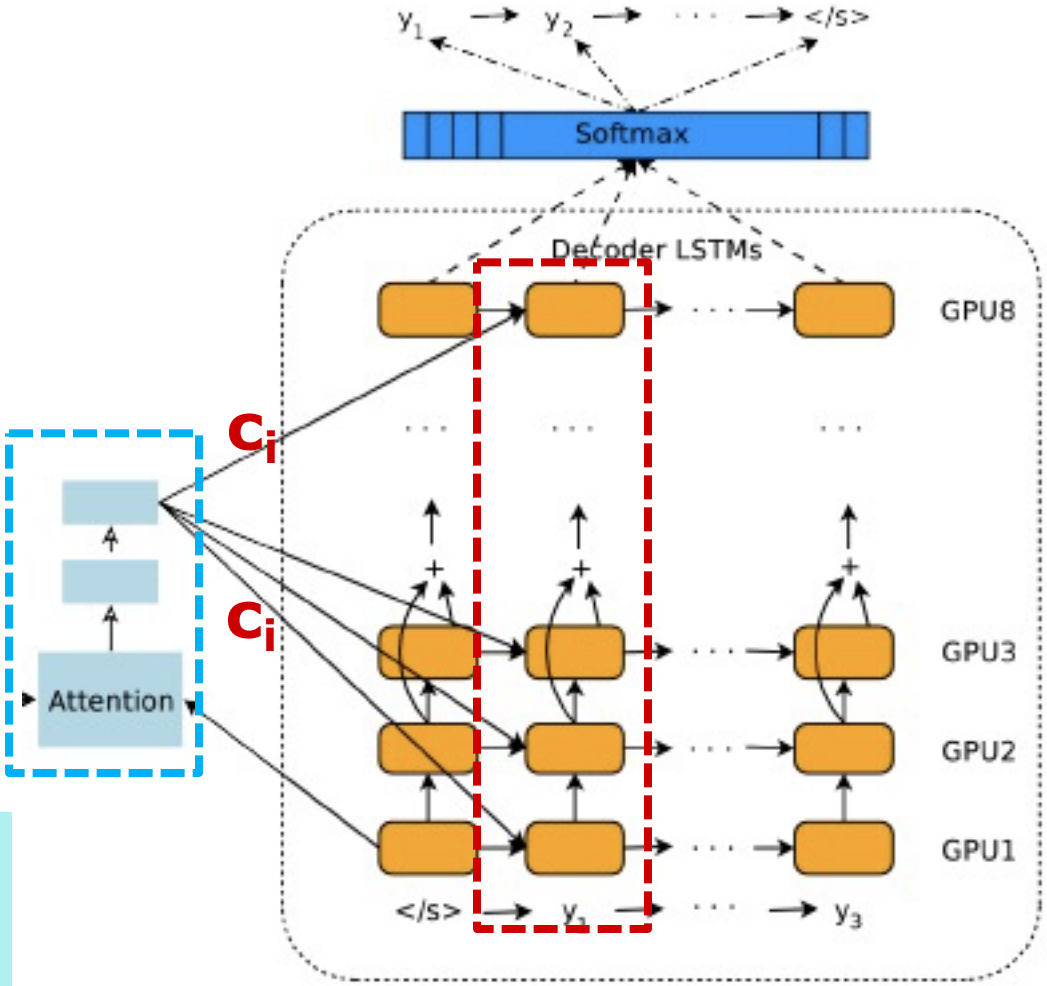
重要な部分は、各デコーダの出力するワード y_tが、Encoderの最後の状態だけでなく、すべての入力状態の重みづけられた結合に依存することである。

aは、出力ごとに、それぞれの入力状態をどの程度考慮されるべきかを定義する重みである。したがって、$a_{3,2}$が大きい場合、これは、Decorderがターゲット文の第3の単語を生成しながら、ソース文の第2の状態に多くの注意を払うことを意味する。 aは、通常、1に合計されるように正規化される（それらは、入力状態に対する確率分布である）。」

Bahdanau et al.



GNMT



Decoderの内部状態 s_iは、
先行するノードの内部状態s_{i-1}と
先行するノードの出力y_{i-1}と、
Context c_iで決まる。

$$s_i = f(s_{i-1}, y_{i-1}, c_i)$$

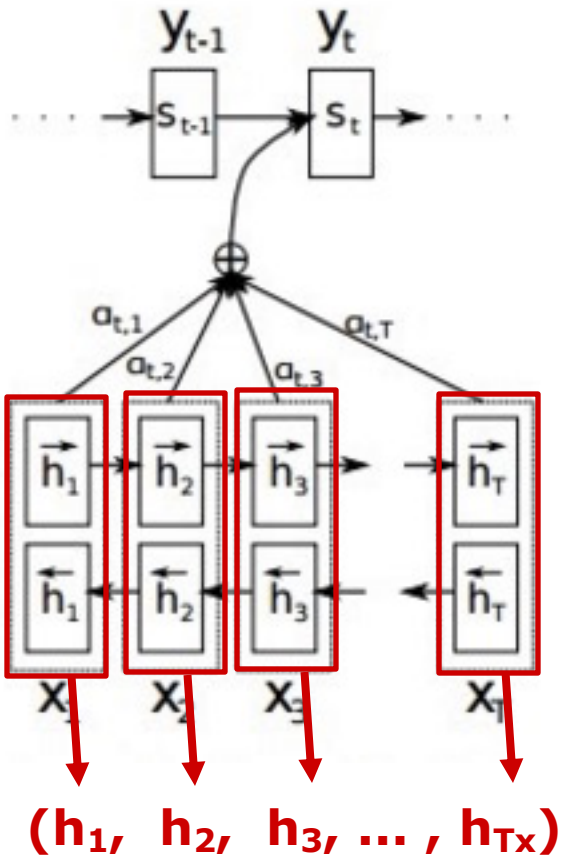
Annotation

このシステムでは、Encoderが、入力のシーケンスをAnnotationのシーケンス
$(h_1, h_2, \ldots, h_{Tx})$ に変える。

先の図には、Annotation h_iの名前は、直接には書き込まれていなかった。

上下に並んだ 逆向きの矢印を持つh_i を囲む四角が書かれているのが、この四角がAnnotation h_i である。ここでは、それを赤い四角で囲んだ。

この論文では、右向きの隠れ層の状態と左向きの隠れ層の状態の「連結」としてh_iが実装されている。GNMTでも同様である。



Context

Docorderが、入力の(x_1, x_2, ... , x_{Tx}) から、t番目の語y_t を生成しようとする時に、このAnnotationのシーケンス (h_1, h_2, ... , h_{Tx}) からの情報は、次の式で、c_i に束ねられてDecoderに流れ込んでくる。

$$c_i = \sum_{j=1}^{T_x} \alpha_{ij} h_j.$$

このc_i を、**Context**と呼んでいる。

Decoderの内部状態

Decoderの内部状態 s_iは、先行するノードの内部状態s_{i-1}と先行するノードの出力y_{i-1}と、このContext c_iで決まる。

$$s_i = f(s_{i-1}, y_{i-1}, c_i)$$

「直観的には、これは、DecoderにAttentionのメカニズムを実装する。Decoderは、ソースの文のいくつかの部分を、注意を払うべき文章だと決定する。Decoderに、Attentionのメカニズムを持たせることで、Encoderは、ソース文内のすべての情報を固定長のベクトルにエンコードする負担から解放される。この新しいアプローチにより、情報は、Annotationのシークエンス上に広がって拡散することができ、その情報は、Decoderによって選択的に取り出すことができる。」

人間と機械翻訳のギャップに橋をかける Googleのニューラル機械翻訳システム

"Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation"

Yonghui Wu et al.

https://goo.gl/YqlAAW

2016年

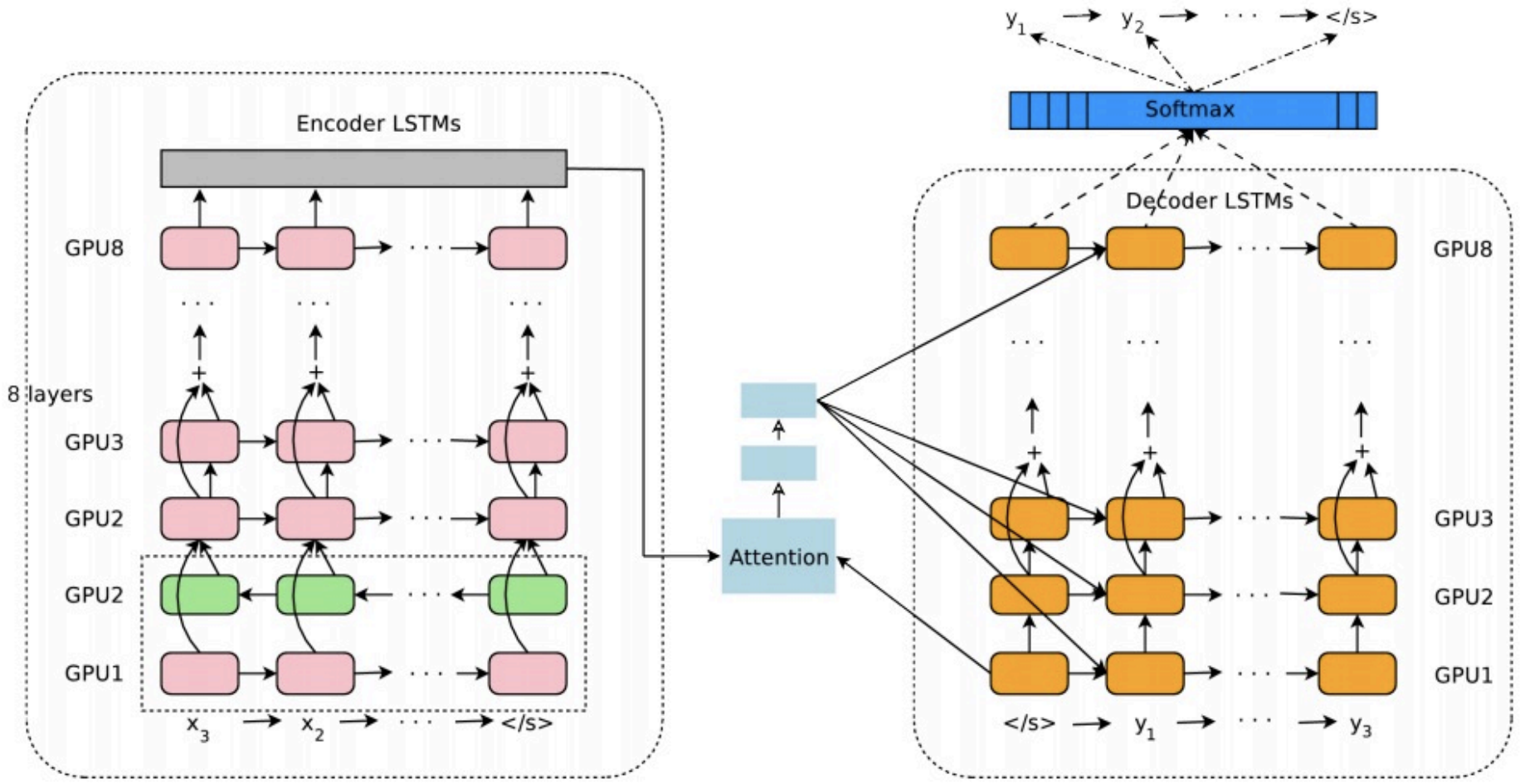


Googleニューラル機械翻訳システムの概観

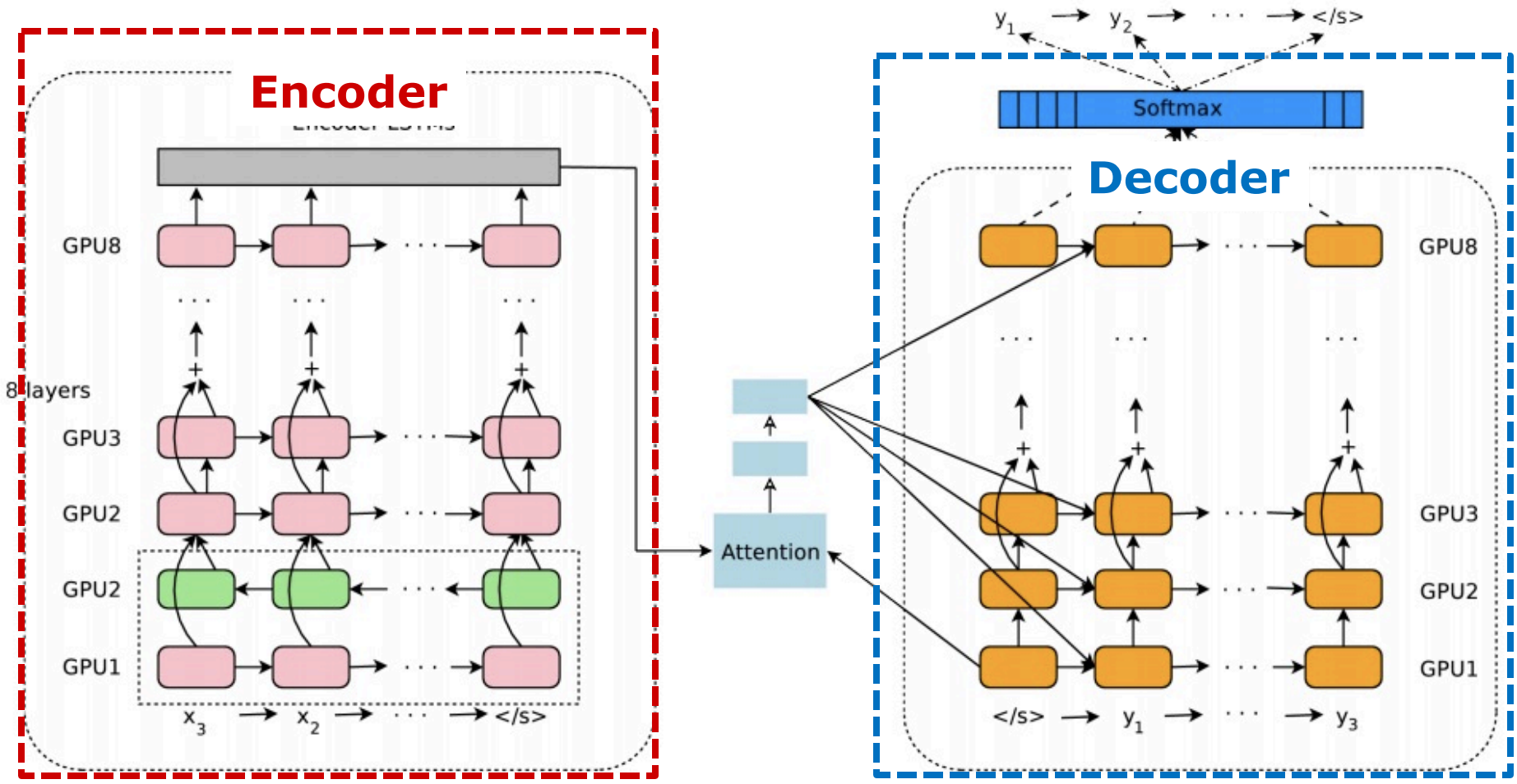
最初に、GNMTのシステムを概観する。

そのいくつかの要素、Residue Connection, Bi-Directional Encoder, Quantification等については、小論では、省略した。

GNMTのアーキテクチャーの概念図

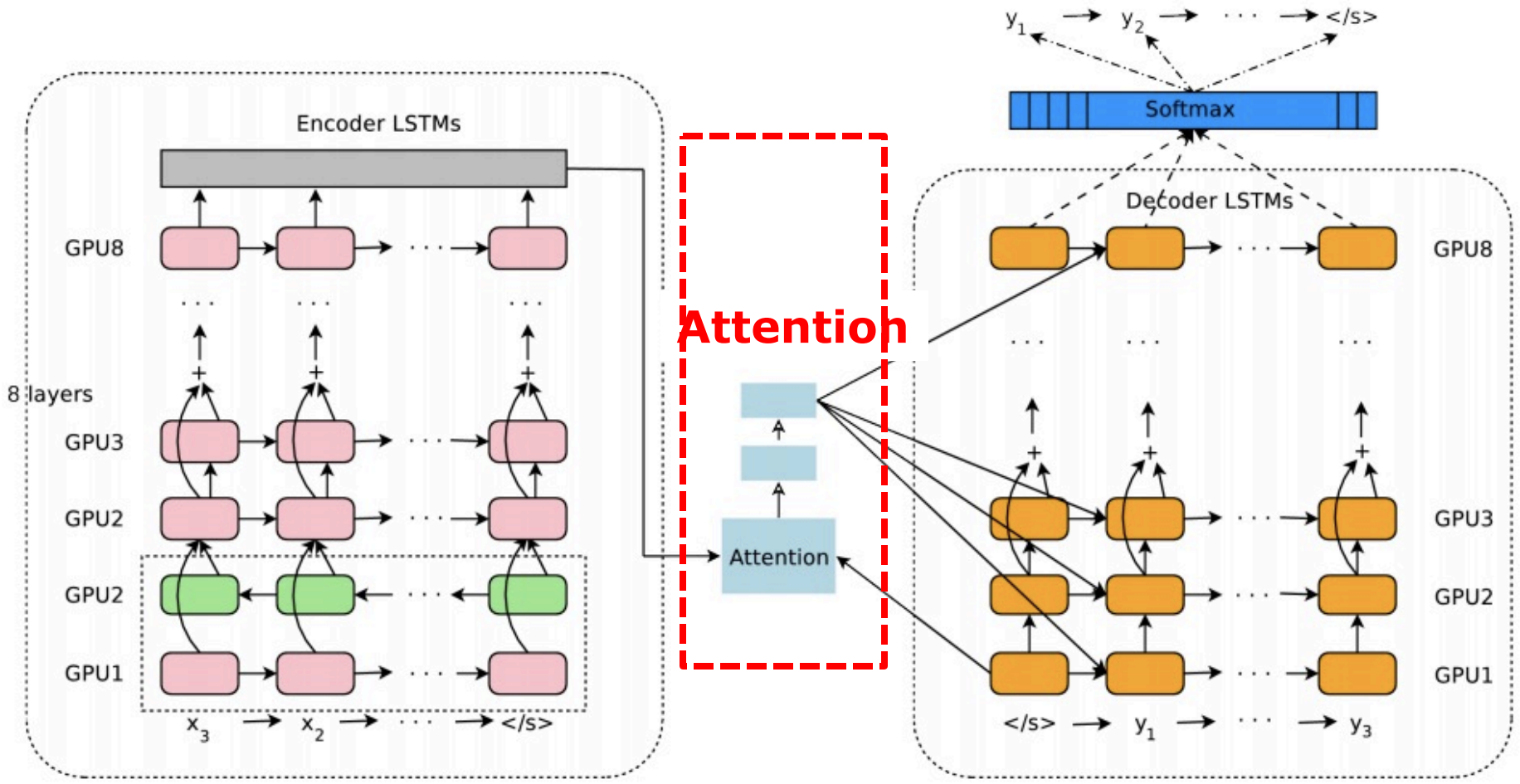


Encoder / Decoder



左側に、LSTMを8段重ねにした Encoder LSTMがあり、
右側には、同じくLSTMを8段重ねにした Decoder LSTMがある。

Attention Mechanism



EncoderとDecoderの中間に、Attentionと記された領域がある。
ここからの出力Attention Context は、Decoderのすべての
ノードに供給されている。

Wordpiece

正直にいうと、僕は最初、論文のこの部分が、何をしているのかよくわからなかった。アプローチが僕の常識の「想定外」だった。その意味では、GNMTらしさが明確に出ているところだと思う。

と言って、言語へのアプローチとして、逸脱があるわけではないのだ。原理的に言えば、言語は、書かれた文字の集まりではなく、まずは、発話された音の連なり（順序は重要）としてあるということと、コトバとその意味の結合は、「恣意的」なものだということにたちかえればいいのだと思う。

この部分は、”Zero-Shot”論文とも、繋がっている。

文をどのように分割するか？

- 機械翻訳の対象は、もちろん、ソース側でもターゲット側でも文なのだが、その文をどのように翻訳機械に入力として与えるのかには、いくつかの選択肢がある。
- 一つは、文を「文字」単位で分割して、1文字ずつ入力することだ。もちろん、文末には、文末であることを示す記号を置く。
- もう一つの方法は、文を「語」に分割して、一語づつ入力をすることだ。ここでも、文末の記号は必要である。

- 1文字ずつ読んで、正しい翻訳ができるのか？　それが出来るのである。次の英語 -> フランス語翻訳の結果は、文字単位の方がワード単位よりいい翻訳結果が得られることを示している。（なぜか？）　ただし、計算時間は、5倍から10倍かかる。

Character Mode

Table 4: Single model results on WMT En→Fr (newstest2014)

Model	BLEU	CPU decoding time per sentence (s)
Word	37.90	0.2226
Character	38.01	1.0530
WPM-8K	38.27	0.1919
WPM-16K	37.60	0.1874
WPM-32K	38.95	0.2118
Mixed Word/Character	38.39	0.2774
PBMT [15]	37.0	
LSTM (6 layers) [31]	31.5	
LSTM (6 layers + PosUnk) [31]	33.1	
Deep-Att [45]	37.7	
Deep-Att + PosUnk [45]	39.2	

同じ文字セットを使っているのなら、一文字単位での「翻訳」は、驚くほど精度は高い！

ただ、効率は悪い

wordへの分割の問題

- ただ、自明と思える文のwordへの分割にも問題があるのだ。(特に、日本語の場合) 次の二つの文を考える。

The stratosphere extends from about 10km to about 50km in altitude.

成層圏は、高度10kmから50kmの範囲にあります。

- 英語の場合、文のワードへの分割は、ワード間にスペースが存在することで、ほぼ自明である。ところが、日本語の場合にはそうではない。
- 文は語から、「文法的」に構成されているのだが、文を語に分割するには、「文法」の知識に基づいたParsingが必要なのだ。（例えば、Mecabを使う）

辞書にない語をどう扱うか？

- ニューラルマシン翻訳モデルでは、名前、数字、日付といった、基本的には辞書にないオープンな語彙の翻訳であっても、固定した語のボキャブラリで動作することがよくある。
- 語彙外（OOV：Out Of Vocaburary ）単語の翻訳に対処するアプローチには、大きく2つのカテゴリーがある。
- 1つのアプローチは、珍しい単語を、ソースからターゲットへの単純にコピーすることである。（珍しい単語の大部分は、名前や番号で、それらに対しては、コピーが正しい翻訳になる。）あるいは、アテンション・モデルに基づいて、外部のアライメントモデルを使う、または、より複雑な特殊な目的のポインティングネットワークを使う方法もある。
- もう一つのアプローチの広いカテゴリーは、サブ・ワード（sub-word）を単位として使うことである。サブ・ワードには、文字（character），語(word)と文字の混在、あるいは、もっと賢いサブ・ワードがある。

ワードピース・モデル

我々の最も成功したアプローチは、第2のカテゴリー（サブワードユニット）に分類される。Google音声認識システムの日本語/韓国語の分割問題を解決するために最初に開発されたワードピースモデル（WPM）[35]の実装を採用した。

このアプローチは、完全にデータ駆動型で、可能性のある文字列に対して確定的な分割を生成することが保証されている。これは、ニューラルマシン翻訳で、辞書にない珍しい言葉を扱うために[38]で使用された方法に似ている。

任意の単語を処理するために、我々はまず、与えられた訓練されたワードピース・モデルでワード達をワードピースに分割する。特別な語境界記号が、モデルの訓練の前に追加される。元のワード列を曖昧さなくワードピース列から復元することができる。デコード時に、モデルはまずワードピース列を生成し、ワードピース列は対応するワード列に変換される。

ワード列とそれに対応するワードピース列の例

- ワード列:
  Jet makers feud over seat width with big orders at stake
- ワードピース列:
  _J et _makers _fe ud _over _seat _width _with _big _orders _at _stake

- 上記の例では、ワード「ジェット」は2つのワードピース「_J」と「et」に分割され、ワード「feud」は2つのワードピース「_fe」と「ud」に分割されている。他のワードは単一のワードピースとして残っている。"_"はワードの先頭を示す特殊文字である。

ワードピース・モデル

- ワードピースモデルは、データドリブン・アプローチを使って生成され、変化するワードの定義の元でも、トレーニングデータの言語モデルの可能性を最大にする。
- 訓練コーパスと、望むトークンの数Dが与えられた時、最適化の問題は、コーパスが選択されたワードピースモデルに従って分割されたとき、結果として得られるコーパスがワードピースの数が最小であるように、D個のワードピースを選択することである。
- この最適化問題に対する我々の総当たりのアルゴリズムは、[38]と同様であり、[35]でより詳細に説明される。

2012年 [35] Schuster, M., and Nakajima, K. ”J**apanese and Korean voice search**” https://goo.gl/DUxQuv

1. まず、単位語の目録を、ベーシックUnicode（日本語の漢字・ひらがな・カタカナ、韓国語のハングル、そしてすべてのASCIIコードを含む）で作る。最終的には、日本語でトータル22000語、韓国語11000の目録になる。
2. この目録を使った訓練データ上に、言語モデルを構築する。
3. 現在の語の目録から二つの単位語を取り出し、一つに結合して新しい単位語を作り、単位語の目録を一つ増やす。可能な組み合わせの中から、言語モデルに追加された時、訓練データ上で、類似度を一番増大させるものを、新しい単位語として選ぶ。
4. 事前に定義された単位語の限界に到達するか、類似度の増大が、ある閾値以下に下がるまで、2. の操作を繰り返す。

- [35]で使用された元の実装と比較して、ワードの始めにのみ特殊記号を使用し、ワードの両端では使用しない。また、基本的な文字の数をデータに応じて管理可能な数に減らし（西洋言語では約500、アジア言語ではより多く）、残った文字を特別な未知の文字にマップして、非常にまれな文字で、ワードピースの語彙を汚染しないようにする。
- トータルで8kないし32kの間のワードピースの語彙を使用すると、我々が試したすべての言語対の間で良好な精度（BLEUスコア）と高速なデコード速度の両方が達成されることがわかった。

Champollionが、"*Je tiens l'affaire!* »と叫んで倒れたのは、象形文字が、表音文字と表意文字の混在であることに気づいたからなのだが、日本語は、もともとそうである。その上、ひらがな、漢字、カタカナ、ASCIIが混在している。（顔文字も）
[35]で、取り上げられている日本語の「読み」の難しさの例が面白い。 "AKB48", "W-INDS", "シャ乱Q", "価格.com", ...

- 前述のように、翻訳では、ソースからターゲットにまれな固有名あるいは番号を直接コピーすることがよくある。このような直接のコピーを容易にするため、ソース言語とターゲット言語の両方に共通のワードピースモデルを使用する。このアプローチを使用すると、ソースとターゲットの文の同じ文字列がまったく同じ方法で分割されることが保証され、システムがこれらのトークンを簡単にコピーできるようになる。
- ワードピースは、キャラクターの柔軟性とワードの効率との間のバランスを達成する。我々のモデルは、ワードピースを使用したときに、BLEUスコア全体が改善されていることがわかる。おそらく、我々のモデルが文字のみに頼らずに本質的に無限のボキャブラリを効率的に扱うからである。文字のみの場合は、入力列と出力列の平均長さをずっと長くするので、より多くの計算が必要になる。

Googleの多言語ニューラル機械翻訳

"Google's Multilingual Neural Machine Translation System: Enabling Zero-Shot Translation"

Melvin Johnson et al.

https://goo.gl/islUXa

2016年



ゼロ・ショット翻訳

この多言語システムは、明示的な訓練データがない言語ペアの間でゼロ・ショット翻訳を実行できる。

著者らは、これを、「我々の知る限り、これは真の多言語ゼロ・ショット翻訳の最初のデモである」と主張している。

インターリンガの存在

言語にかかわらず、ネットワークは、同じ意味を持つ文章が同じような方法で表現される何らかの共有表現を学習しているのか？　この質問に対しては、そうだという。これも、画期的。

モデルは、訓練された言語ペアを扱うのと同じ方法で、ゼロ・ショット翻訳を実行しているのか？　これについては、まだよくわからないという。

2016年 Melvin Johnson et al. ***"Google's Multilingual Neural Machine Translation System: Enabling Zero-Shot Translation"***
https://goo.gl/islUXa

「我々は、単一のニューラル機械翻訳（NMT）モデルを使用して、複数の言語どうしを翻訳する、シンプルで洗練されたソリューションを提案する。」

「共有ワードピースのボキャブラリを使用することで、多言語NMTはパラメータを増やさずに、単一のモデルを利用することができる。これは、従来の多言語NMTの提案よりも大幅に簡単なものだ。」

「我々のモデルは、訓練中に明示的には見られなかった言語ペアの間の暗黙的な橋渡しを実行することも学ぶことができた。それは、翻訳の学習とゼロ・ショット翻訳がニューラル翻訳で可能であることを示している。」

「我々のモデルには、普遍的なインターリンガ表現が存在することを示唆する分析を示し、複数の言語を混在させた時に起きる、興味深い例を示す。」

Google 多言語ニューラル機械翻訳の概要

多言語の翻訳を、言語ペアの数だけのシステムによって行うのではなく、一つのシステムで行うことにより、さらに多くの言語へのスケール・アップが容易になる。

システムが単純になり、コーパスの少ない言語にもメリットが生まれ、ゼロ・ショット翻訳も可能になる。

多言語翻訳を単一モデルで

- ニューラル機械翻訳（NMT）は、多くの大規模な環境で急速に採用されてきた機械翻訳に対するアプローチである。ただ、このようなシステムのほとんどは、単一の言語のペアのために構築されていた。
- 基本的なNMTアーキテクチャを大幅に変更することなく、単一のモデルを使用して複数言語のペアを処理するための十分に単純で効率的な方法は、それまでなかった。
- この論文では、単一のモデルを使用して多言語間で翻訳を行うための簡単な方法を紹介している。この方法では、ターゲット言語を示す人工的なトークンを、入力シーケンスに追加するだけで、従来のNMTモデルアーキテクチャに、変更を加える必要はない。

特徴1： シンプルさ

- モデルのアーキテクチャは変更されていないため、新しいデータを単純に追加するだけで、より多くの言語へのスケーリングは簡単に行われる。
- ターゲット言語が変更された場合には、それを示す新しいトークンが使用されるだけである。これにより、複数の言語を扱う際に必要なモデルの数を減らすことができるため、製品版の展開も簡単なものになる。
- Googleでは、ソースとターゲットとして合計100以上の言語をサポートしているため、もし、それぞれの翻訳のモデルが、1つの言語ペアしかサポートしていないとすれば、すべての言語ペア間の最良の翻訳を得るためには、理論的には、100^2ものモデルが必要になる。明らかに、これは製品版の環境では問題となる。

特徴2： リソースの少ない言語の翻訳の改善

- 多言語NMTモデルでは、すべてのパラメータが、モデル化されているすべての言語ペアによって暗黙のうちにに共有される。これにより、モデルは、訓練・学習を通じて言語の境界を越えて一般化される。
- 利用可能なデータがほとんどない言語のペアと豊富なデータを持つ言語のペアが単一のモデルに混在すると、リソースの少ない言語ペアの翻訳品質が大幅に向上する。

フランス語 -> 日本語 で翻訳してみたのだが、
フランス語 -> 英語 -> 日本語 のブリッジの方が
精度が高いように思われる。

特徴3：ゼロ・ショット翻訳

- 単一のモデルで複数の言語ペアをモデリングすることの驚くべきメリットは、モデルが、これまで見たことのない言語ペア間の翻訳（ゼロ・ショット翻訳）を暗黙のうちに学習することである。
- たとえば、ポルトガル語から英語、英語からスペイン語のサンプルで訓練された多言語NMTモデルでは、その言語ペアの学習データは存在しないにもかかわらず、ポルトガル語からスペイン語への合理的な翻訳が生成される。
- 問題の言語ペアの追加データをほとんど使わずに、ゼロ・ショット言語ペアの品質を簡単に改善できる。

訓練に利用されたコーパス

- WMT'14の英語(En)<-> フランス語(Fr)データセットには、3,600万の文のペアが含まれている。
- WMT'14の英語(En) <-> ドイツ語(De)データセットには、500万の文のペアが含まれている。
- Googleは、内部に、英語 <-> 日本語(Ja)、英語 <-> 韓国語(Ko)、英語 <-> スペイン語(Es)、英語 <-> ポルトガル語(Pt) 等々の多くのデータセットを持っているが、その規模は、先のWMTのデータセットより、2～3桁大きいという。

- Googleニューラル機械翻訳では、GPU100個を使って、フルトレーニングには最大1,000万ステップ、収束までには3週間かかることがあるという。

ビジュアルな分析
インターリンガの存在

言語にかかわらず、ネットワークは、同じ意味を持つ文章が同じような方法で表現される何らかの共有表現を学習しているのか？　この質問に対しては、そうだという。これも、画期的。

モデルは、訓練された言語ペアを扱うのと同じ方法で、ゼロ・ショット翻訳を実行しているのか？　これについては、まだよくわからないという。

実験結果が提起する問題

- モデルを複数の言語にまたがってトレーニングすることで、個々の言語レベルでのパフォーマンスが向上し、ゼロ・ショット翻訳が有効になることがわかるということが、この論文の結論なのだが、この結果は、これらのタスクがモデル内でどのように処理されるかについて、次のような問題を提起する。

1. 言語にかかわらず、ネットワークは、同じ意味を持つ文章が同じような方法で表現される何らかの共有表現を学習しているのか？
2. モデルは、訓練された言語ペアを扱うのと同じ方法で、ゼロ・ショット翻訳を実行しているのか？

ビジュアル化による分析

- ネットワークによって使用される表現を研究する1つの方法は、翻訳中のネットワークの活性化の状態を見ることである。
- 研究の出発点は、アテンション・ベクトルのセット、すなわち、エンコーダとデコーダのネットワークを接続するレイヤ内の活性化の状況を見ることだ。
- 単一の文の翻訳は、一連のアテンション・ベクトルを生じさせる。この文脈において、共有表現に関する元々の問題は、異なる文のベクトルのシーケンスがどのように関連しているかを調べることで研究することができる。例えば、次のように。
- ソースまたはターゲット言語に応じて、文章がクラスタリングされているのか？
- あるいは、言語にかかわらず、同様の意味を持つ文章がクラスタリングされるのか？

インターリンガの証拠

- いくつかの訓練されたネットワークは、実際に共有表現の強いビジュアルな証拠を示す。
- たとえば、以下の図2は、英語 <-> JapaneseとEnglish <-> Koreanでトレーニングされた多対多モデルから作成されたものである。モデルの実際の動作を視覚化するために、意味論的に同一の言語間フレーズの74個のトリプルからなる小さなコーパスから始めた。
- つまり、それぞれのトリプルには、同じ基本的な意味を持つ英語、日本語、韓国語のフレーズが含まれている。これらのトリプルをコンパイルするために、私たちは日本語と韓国語の翻訳と対になった、英語の文章のための正しい（Ground Truth）データベースを検索した。

英語

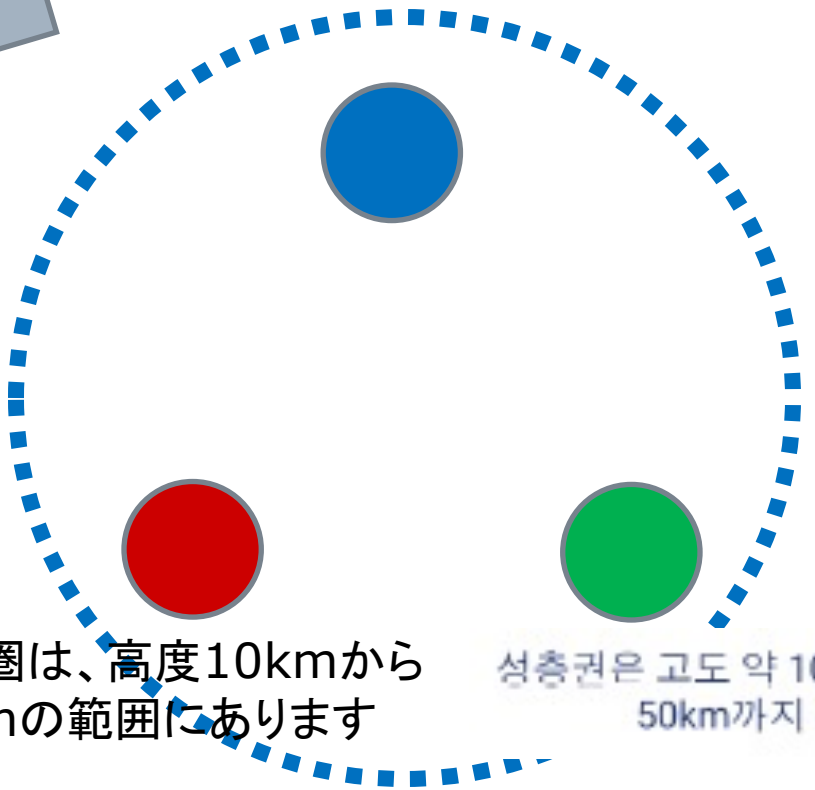
三つの言語で
同じ意味を持つ文を
一つのトリプルに
まとめる。

日本語

韓国語

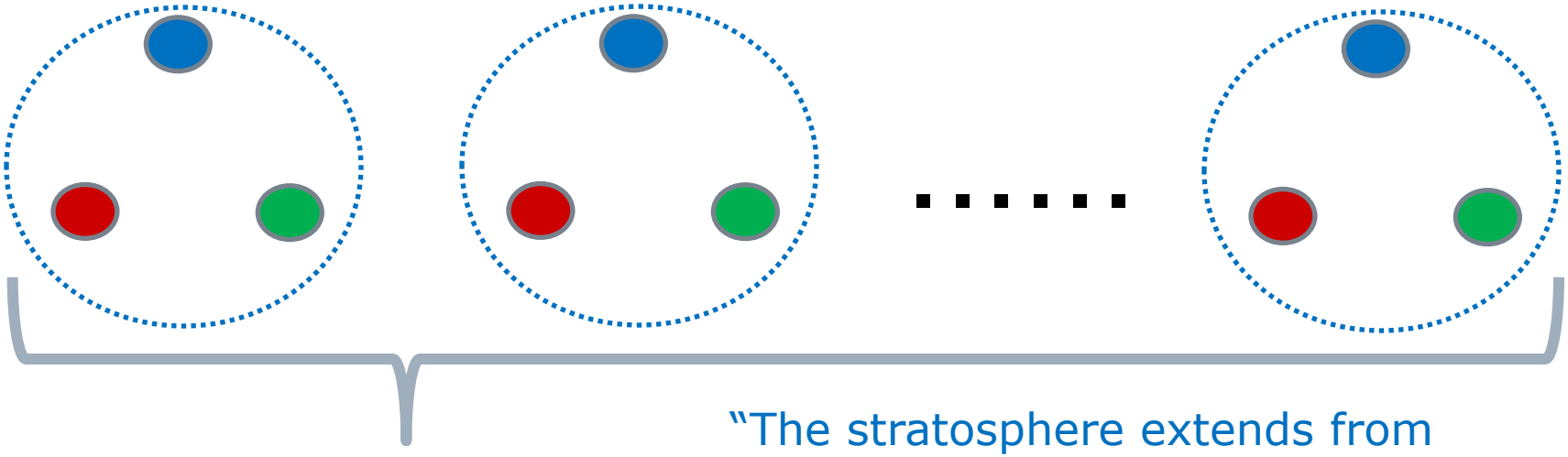
例えば

“The stratosphere extends from about 10km to about 50km in altitude.”



成層圏は、高度10kmから
50kmの範囲にあります

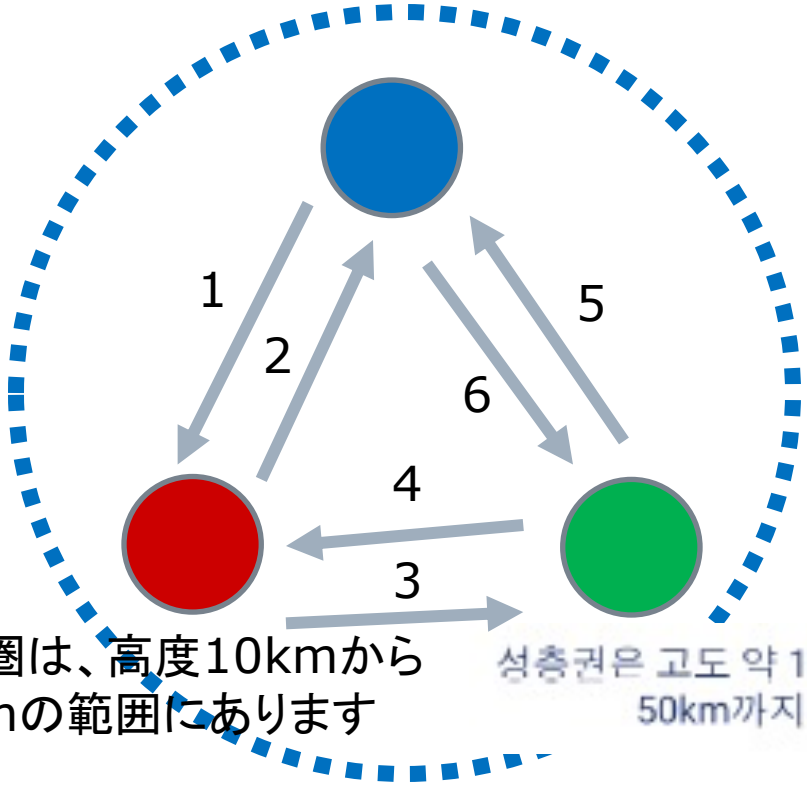
성층권은 고도 약 10km부터 약 50km까지 확장됩니다.



74個のトリプルからなる
コーパスを準備する。

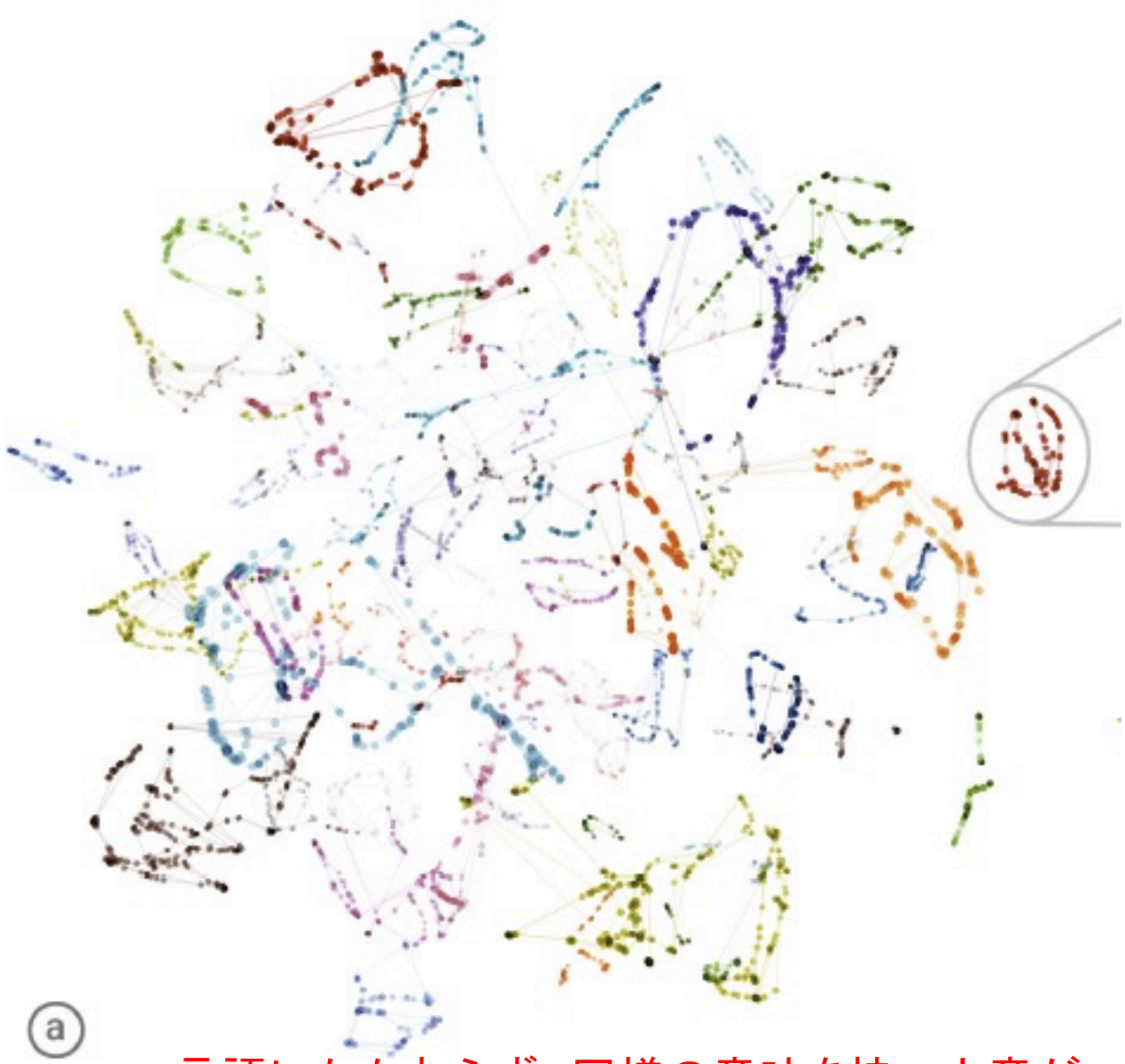
"The stratosphere extends from about 10km to about 50km in altitude."

各トリプルの各センテンスを
他の2つの言語に翻訳する。
一つのトリプルについて、
6つの翻訳が生まれる。



このコーパスでは、
74 x 6 = 444 の
翻訳が生まれる。

成層圏は、高度10kmから
50kmの範囲にあります



言語にかかわらず、同様の意味を持つ文章がクラスタリングされている。

この 444の翻訳を行うのに、システムは9,978ステップ要した。

この時、発せられた9,978個のアテンションベクトルを、t-SNEを用いて二次元に投射したのが左の図。

同じトリプル内の文の翻訳で発せられるアテンション・ベクトルは、同じ色で表している。

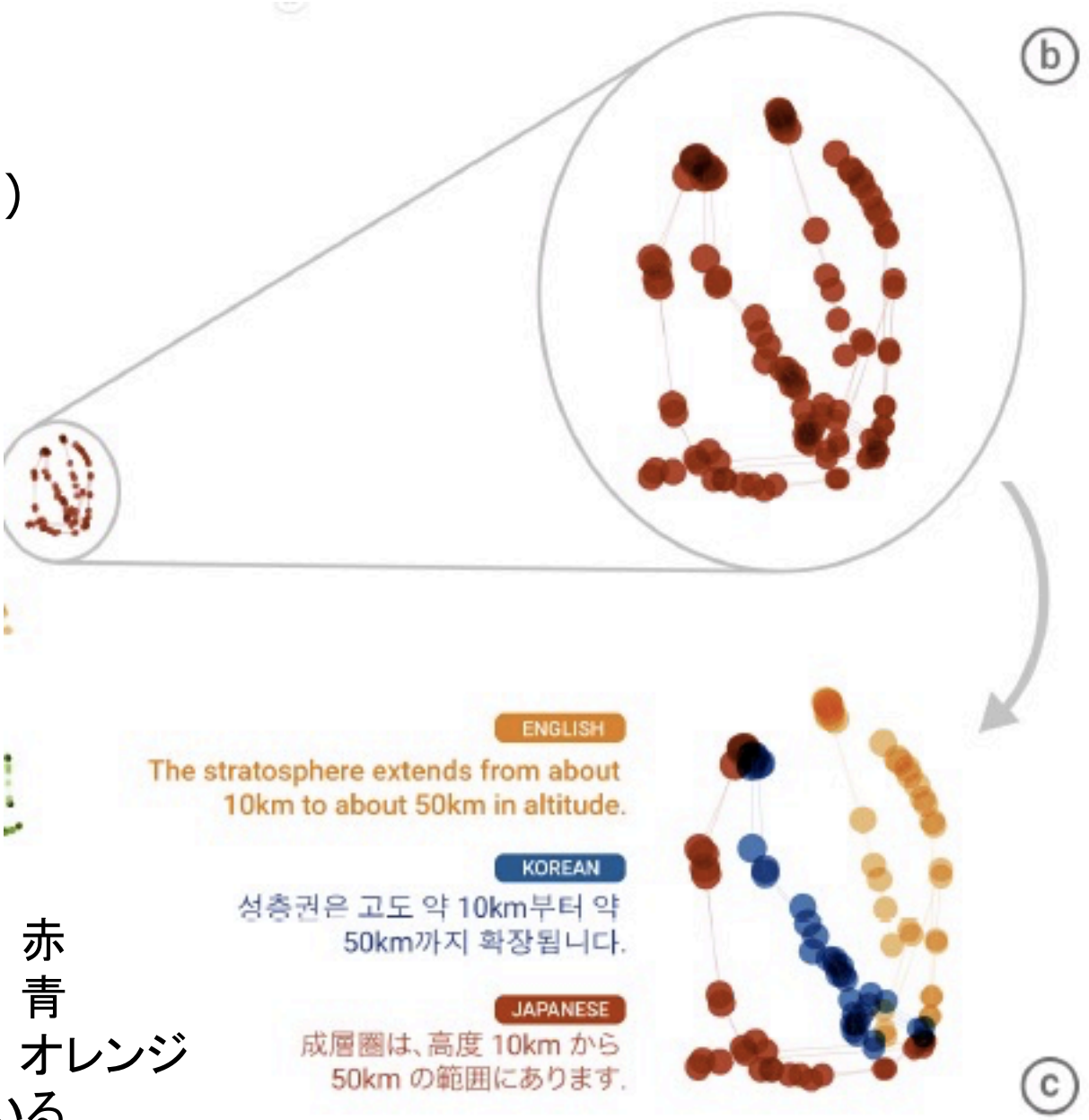
同じ色が、近くに集まってクラスターを形成しているのが見て取れる。（？）

この同じ色のクラスター(b)は、元の同じ意味を持つトリプルから生まれたものである。

同一の文から発せられるアテンション・ベクトルは、線で結ばれている。

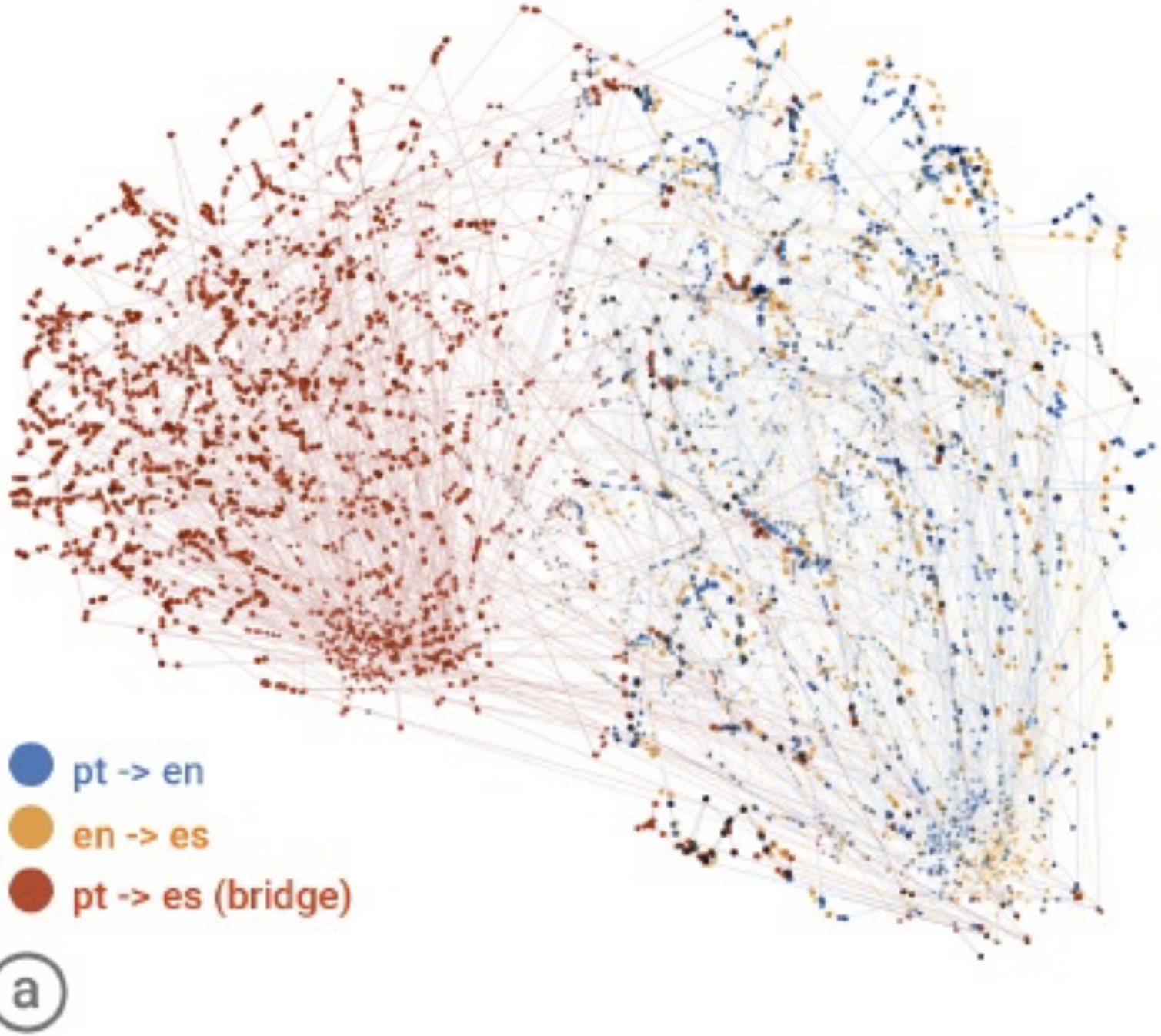
(c)は、翻訳ソースが同じ言語を同じ色で塗り分けたもの。

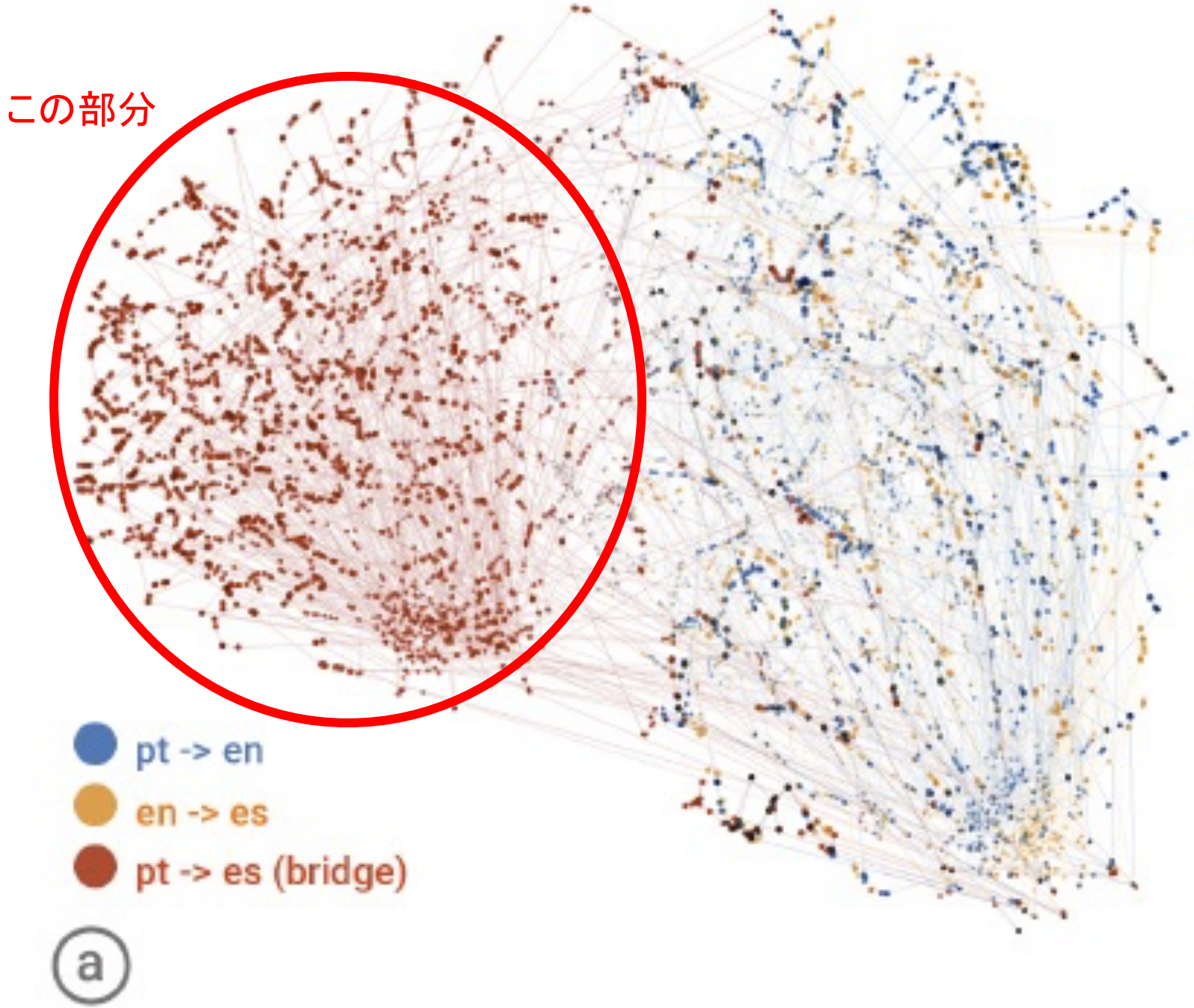
日本語->{英語,韓国語} 赤
韓国語->{英語,日本語} 青
英語->{日本語,韓国語} オレンジ
は、クラスターを形成している。



部分的に分離された表現

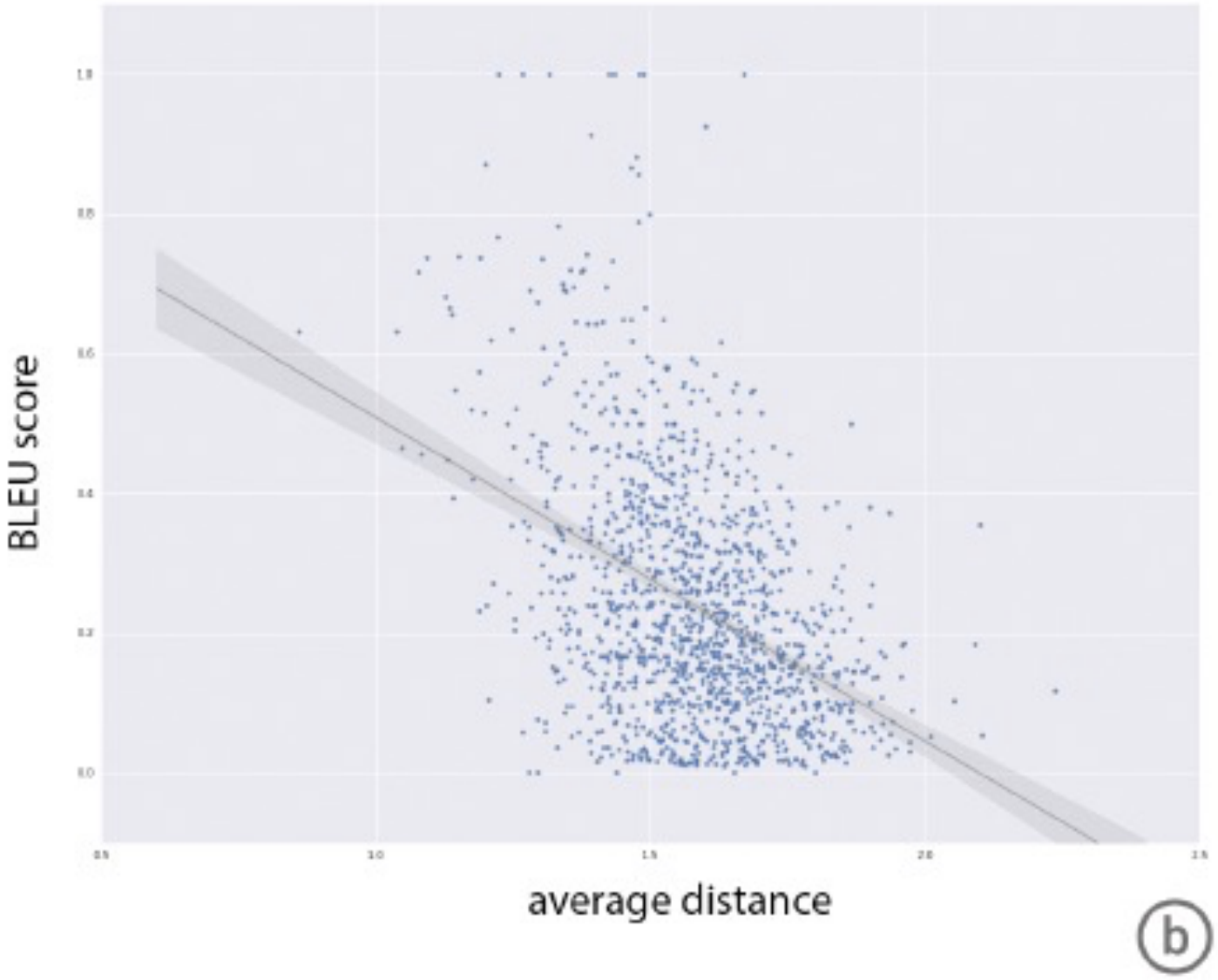
- すべてのモデルが、そのようなクリーンな意味クラスタリングを示すわけではない。ある一つの言語ペアからだけの多くのアテンション・ベクトルを含む、大きなクラスター観察した。
- たとえば、次の図は、ポルトガル語 -> 英語（青）と英語 -> スペイン語（黄色）で訓練され、ポルトガル語からスペイン語（赤）のゼロショット変換を実行したモデルからの注意ベクトルのt-SNEプロジェクションを示しています。
- 左の大きな赤い領域には、主にゼロショットのポルトガル語 -> スペイン語の翻訳が含まれている。言い換えれば、かなりの数の文では、ゼロ・ショット翻訳は、2つの訓練された翻訳方向とは異なる埋め込みを持っている。


pt -> en
en -> es
pt -> es (bridge)
a


この部分
pt -> en
en -> es
pt -> es (bridge)
a

- 「分離された」ゼロ・ショット変換の大きなクラスターが、何らかの重要性を持っているかどうかを尋ねるのは自然なことだ。
- 確定的な回答にはさらなる調査が必要なのだが、この場合、分離された領域のゼロ・ショット翻訳ではBLEUスコアが低くなる傾向がある。（次の図） この相違度スコアの値はピアソン相関係数が-0.42のゼロショット変換の品質と相関があり、中程度の相関を示している。
- 将来の研究のための興味深い領域は、訓練された言語ペアを介した翻訳の埋め込みと比較することによって、デコーダ内のゼロショット変換の品質を予測するために、埋め込みの地図とモデル性能との間のより信頼できる対応を見つけることである。

ゼロ・ショット翻訳のBLEUスコアと
ゼロ・ショット翻訳とノンブリッジ翻訳
との平均点距離の散布図



混合言語

多言語モデルで、文章の途中で言語を切り替えて混在させるとどうなるか？　訓練データ中には、そうしたチャンポンの例はないにもかかわらず。正しい翻訳が得られる。

出力をチャンポンにできるか？　それは難しかった。

言語の混合

- 追加のソーストークンを使用してランダムなソース言語から選択した単一のターゲット言語に翻訳する仕組みを持っているので、ソースまたはターゲット側で言語が混在しているときに何が起こるかを考え、ました。次の2つの実験を行った。

1. 多言語モデルは、文章の途中での多言語入力（コード切り替え）をうまく処理できるか？

2. 多言語モデルがトリガーされ、1つではなく2つのターゲット言語トークンが重み付けされて1つに加わる（これらのトークンの重み付けされた埋め込みをマージするのと同等）場合、どうなるか？

ソース言語でのコード転換

- ここでは、{日本語、韓国語} ->英語 データで訓練された多言語モデルの例を示す。このモデルを使用して、ソースに日本語と韓国語を混在させると、このモデルではコード切り替えが処理できる。
- そのようなコード切り替えサンプルはトレーニングデータには存在しないにもかかわらず、多くの場合正しい英語の翻訳が生成される。
- モデルには、個々の文字/ワードピースが語彙ボキャブラリー内に存在するため、異なるタイプの記述を効果的に処理できるのだ。

- **Japanese:** 私は東京大学の学生です。 → I am a student at Tokyo University.
- **Korean:** 나는 도쿄 대학의 학생입니다. → I am a student at Tokyo University.
- **Mixed Japanese/Korean:** 私は東京大学학생입니다. → I am a student of Tokyo University.

重みづけられたターゲット言語の選択

- エンコーダLSTMの最下層に〝<2ja>”の埋め込みベクトルを与えるのではなく、線形結合(1-w)<2ja>　＋ w <2ko>を与える。
- 明らかに、w ＝ 0の場合、モデルは日本語を生成する必要があり、w ＝ 1の場合、それは韓国語を生成するはずだが、wがその間の値をとる時、何が起こるのかということ。
- 結果はそれほど驚くことではない。たいていの場合、出力はw ＝ 0.5付近で言語を切り替える。場合によっては、wの中間の値について、モデルは文の途中で、言語を切り替える。
- この動作の可能な説明は、デコーダLSTMによって暗黙的に学習されたターゲット言語モデルが、特にこれらの言語が異なった記述を使用する場合、異なる言語の単語を混在させることを非常に困難にする可能性があるということ。

Russian/Belarusian:	I wonder what they'll do next!
$w_{be} = 0.00$	Интересно, что они сделают дальше!
$w_{be} = 0.20$	Интересно, что они сделают дальше!
$w_{be} = 0.30$	Цікаво, что они будут делать дальше!
$w_{be} = 0.44$	Цікаво, що вони будуть робити далі!
$w_{be} = 0.46$	Цікаво, що вони будуть робити далі!
$w_{be} = 0.48$	Цікаво, што яны зробяць далей!
$w_{be} = 0.50$	Цікава, што яны будуць рабіць далей!
$w_{be} = 1.00$	Цікава, што яны будуць рабіць далей!

ウクライナ語が出てくる！

Japanese/Korean:	I must be getting somewhere near the centre of the earth.
$w_{ko} = 0.00$	私は地球の中心の近くにどこかに行っているに違いない。
$w_{ko} = 0.40$	私は地球の中心近くのどこかに着いているに違いない。
$w_{ko} = 0.56$	私は地球の中心の近くのどこかになっているに違いない。
$w_{ko} = 0.58$	私は지구の中心의가까이에어딘가에도착하고있어야한다。
$w_{ko} = 0.60$	나는지구의센터의가까이에어딘가에도착하고있어야한다。
$w_{ko} = 0.70$	나는지구의중심근처어딘가에도착해야합니다。
$w_{ko} = 0.90$	나는어딘가지구의중심근처에도착해야합니다。
$w_{ko} = 1.00$	나는어딘가지구의중심근처에도착해야합니다。

Spanish/Portuguese:	Here the other guinea-pig cheered, and was suppressed.
$w_{pt} = 0.00$	Aquí el otro conejillo de indias animó, y fue suprimido.
$w_{pt} = 0.30$	Aquí el otro conejillo de indias animó, y fue suprimido.
$w_{pt} = 0.40$	Aquí, o outro porquinho-da-índia alegrou, e foi suprimido.
$w_{pt} = 0.42$	Aqui o outro porquinho-da-índia alegrou, e foi suprimido.
$w_{pt} = 0.70$	Aqui o outro porquinho-da-índia alegrou, e foi suprimido.
$w_{pt} = 0.80$	Aqui a outra cobaia animou, e foi suprimida.
$w_{pt} = 1.00$	Aqui a outra cobaia animou, e foi suprimida.

Google翻訳は、意味を理解しているのか？

- 注意すべきなのは、Googleの「ニューラル機械翻訳」で、RNNが、言語の意味理解の能力を獲得したわけではないだろうということ。
- ある言語Aを母語とするある人が、ある「意味」を込めた文S_Aを発話したとする。もちろんこの文は、A言語の文法にかなっている。機械翻訳システムは、このS_Aを他の言語Bの文S_Bに変換する。もしこの文S_Bが、B言語の文法にかなっていれば、B言語を母語とする人は、この文S_Bに「意味」を見い出す。
- 翻訳システムは、S_Aの意味をS_Bの意味に翻訳したように見えるのだが、そう見えるのは、送り手と受け手の双方の「人間」が、それぞれの文の意味の解釈を、行なっているからである。

Google翻訳は、意味を理解しているのか？

- 翻訳システムが行なっているのは、A言語の文法的な文字列S_Aを、「対応する」B言語の文法的な文字列S_Bに、書き換えているだけである。そこには、「意味」の介在は必要ではない。
- 問題は、「対応」の中身だが。一番、自然な解釈は、A言語の文法とB言語の文法の「対応」である。
- 更に言えば、もしも、このシステムが、基本的には同一の構成のままで、A言語、B言語だけでなく、多言語間の翻訳が可能であるなら、このシステムは、多くの言語の文法の対応付けを可能とする、より深い共通の文法構造を理解していると考えることができる。これは、Chomskyの言う、**Universal Grammer** 普遍文法だと思うと、興味深い。

自然言語理解と機械翻訳

- 機械翻訳では、人間と人間のあいだに機械が介在する。その構造は、機械翻訳のシステムがいかに変わっても、変わることがない。このシステムは、両端にいる人間の言語能力によって支えられている。機械翻訳の精度がいかに向上しても、両端の人間が、当該の言語を理解できなければ、このシステムは機能しない。
- ただ、人間が介在しない、機械翻訳のシステムも存在する。コンピュータのコンパイラーが、まさにそうだ。話が違うと思われたかもしれないが、二つの機械翻訳システムの比較は、実は、いろいろ示唆的である。（人工的な言語と自然言語の違い。コンパイラーが、Googleニューラル機械翻訳の流儀で、訓練され実行されたら、何が起きるかとか。僕は、そんなコンパイラー使いたくない。）

```
/*
 * Increment the size file of the new incorrect UI_FILTER group information
 * of the size generatively.
 */
static int indicate_policy(void)
{
 int error;
 if (fd == MARN_EPT) {
  /*
   * The kernel blank will coeld it to userspace.
   */
  if (ss->segment < mem_total)
   unblock_graph_and_set_blocked();
  else
   ret = 1;
  goto bail;
 }
 segaddr = in_SB(in.addr);
 selector = seg / 16;
 setup_works = true;
 for (i = 0; i < blocks; i++) {
  seq = buf[i++];
  bpf = bd->bd.next + i * search;
  if (fd) {
   current = blocked;
  }
```

Karpathy
http://goo.gl/mNqwCv

RNNが産み出した
Cプログラム・モドキ

Linuxのソースコードを
「学習」させたもの

「中国人の部屋」問題

- ある小部屋の中に、アルファベットしか理解できない人を閉じこめておく（例えば英国人）。この小部屋には外部と紙きれのやりとりをするための小さい穴がひとつ空いており、この穴を通して英国人に1枚の紙きれが差し入れられる。そこには彼が見たこともない文字が並んでいる。これは漢字の並びなのだが、英国人の彼にしてみれば、それは「★△◎▽☆□」といった記号の羅列にしか見えない。 彼の仕事はこの記号の列に対して、新たな記号を書き加えてから、紙きれを外に返すことである。どういう記号の列に、どういう記号を付け加えればいいのか、それは部屋の中にある1冊のマニュアルの中に全て書かれている。例えば"「★△◎▽☆□」と書かれた紙片には「■@◎▽」と書き加えてから外に出せ"などと書かれている。

https://goo.gl/QRPy2e

「中国人の部屋」問題

- 彼はこの作業をただひたすら繰り返す。外から記号の羅列された紙きれを受け取り（実は部屋の外ではこの紙きれを"質問"と呼んでいる）、それに新たな記号を付け加えて外に返す（こちらの方は"回答"と呼ばれている）。すると、部屋の外にいる人間は「この小部屋の中には中国語を理解している人がいる」と考える。しかしながら、小部屋の中には英国人がいるだけである。彼は全く漢字が読めず、作業の意味を全く理解しないまま、ただマニュアルどおりの作業を繰り返しているだけである。それでも部屋の外部から見ると、中国語による対話が成立している。

Searle, John. (1980). “Minds, Brains, and Programs.” https://goo.gl/KvonJi

https://goo.gl/QRPy2e

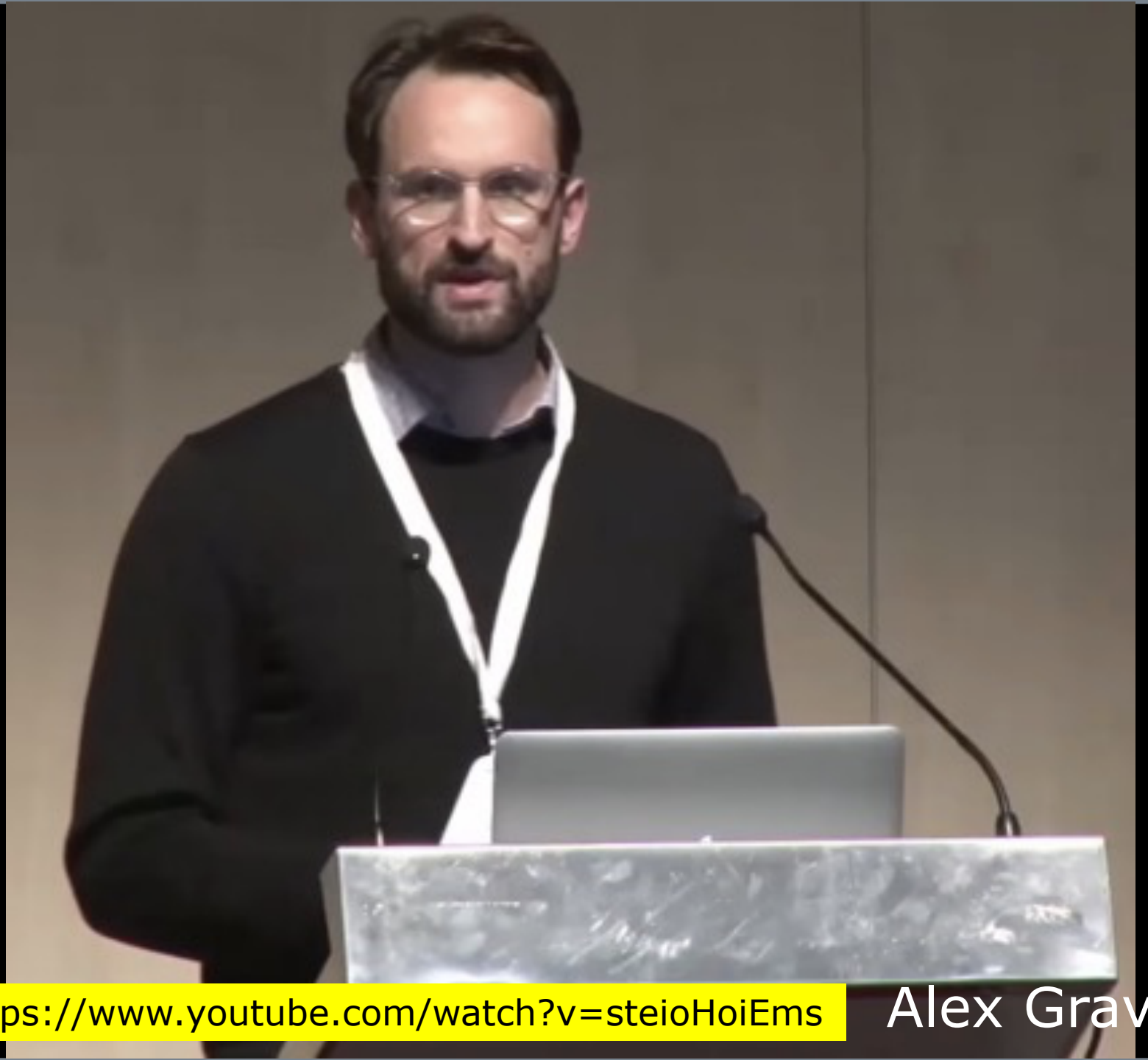
ボイス・アシスタント・システムの実装との類似

- この「中国人の部屋」問題は、言語理解におけるチューリングテストの相似物だと考えていい。部屋の中に機械があるのではなく人間が入っているという想定。
- ただ、中の人間がやっている作業は、マニュアルを引いて答えを返す（これは「検索」だ）だけなので、機械でもできるはず。中の人間を検索する機械に置き換えても、同じような問題は成立する。そして、それは、現在のボイス・アシスタント・システムの実装に近いものになる。このシステムは、言語を理解しているのであろうかと考えてみれば良い。
- それは、「マニュアル」の出来・不出来に依存するはず。それでは、誰が「マニュアル」を作るのか？人間か機械か、はたまた、中に人間が入っている機械か？　運が悪いと、玉ねぎの皮をむくように「無限退行」が始まる。

"Hybrid computing using a neural network with dynamic external memory"

「動的な外部メモリーを備えたニューラル・ネットワークを利用したハイブリッド・コンピューティング」

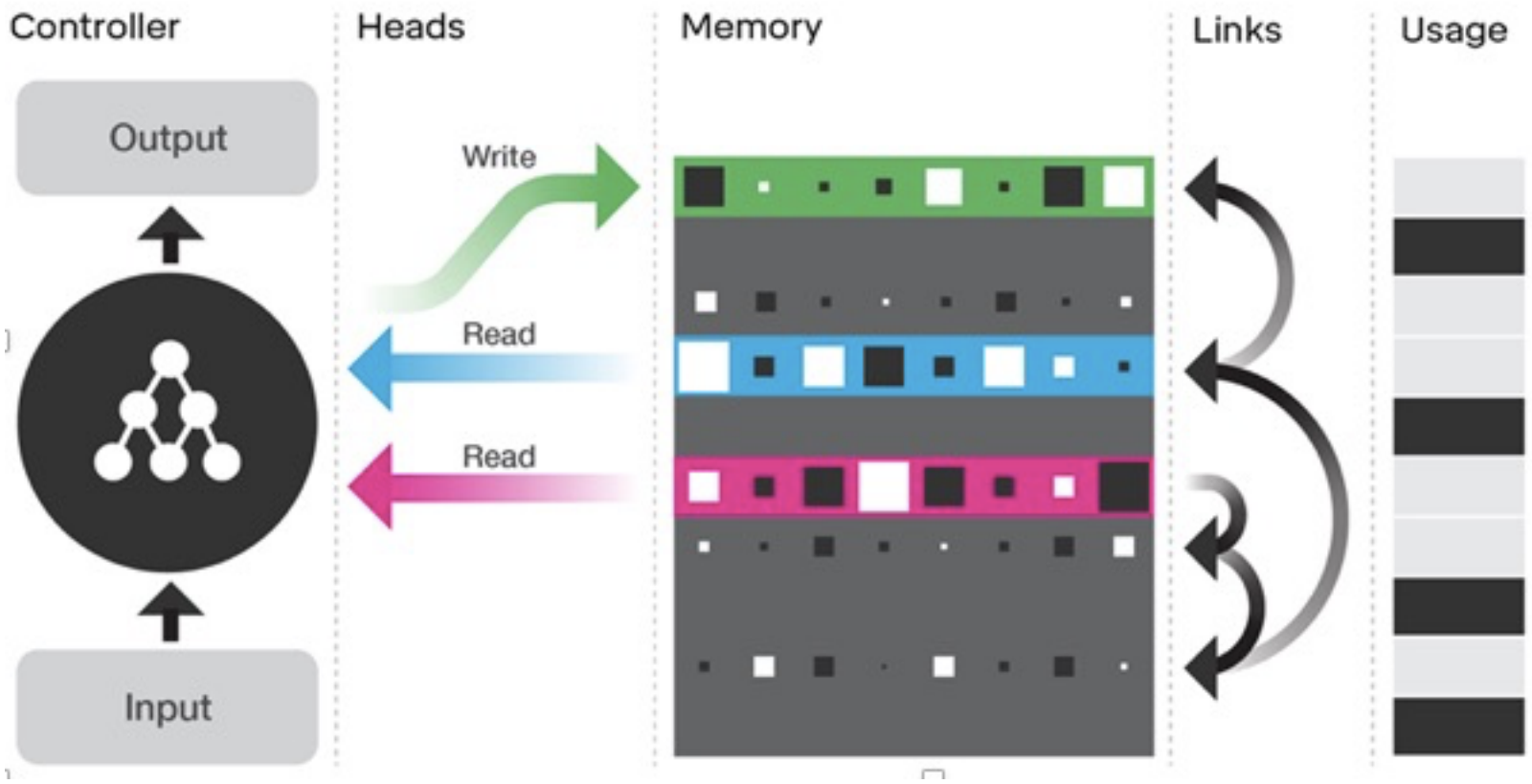
Alex Graves et al. https://goo.gl/r4g5xt



https://www.youtube.com/watch?v=steioHoiEms

Alex Graves

Differentiable Neural Computer
by Google Deep Mind



Synthetic question answering experiments

John is in the playground.
Bob is in the office.
Where is John? A:playground

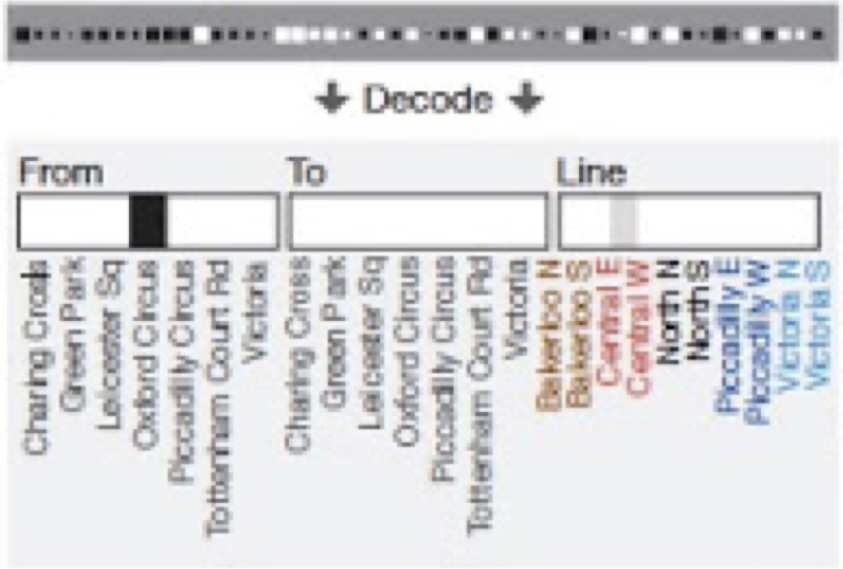
John is in the playground.
Bob is in the office.
John picked up the football.
Bob went to the kitchen.
Where is the football? A:playground
Where was Bob before the kitchen? A:office

John picked up the apple.
John went to the office.
John went to the kitchen.
John dropped the apple.
Where was the apple before the kitchen? A:office

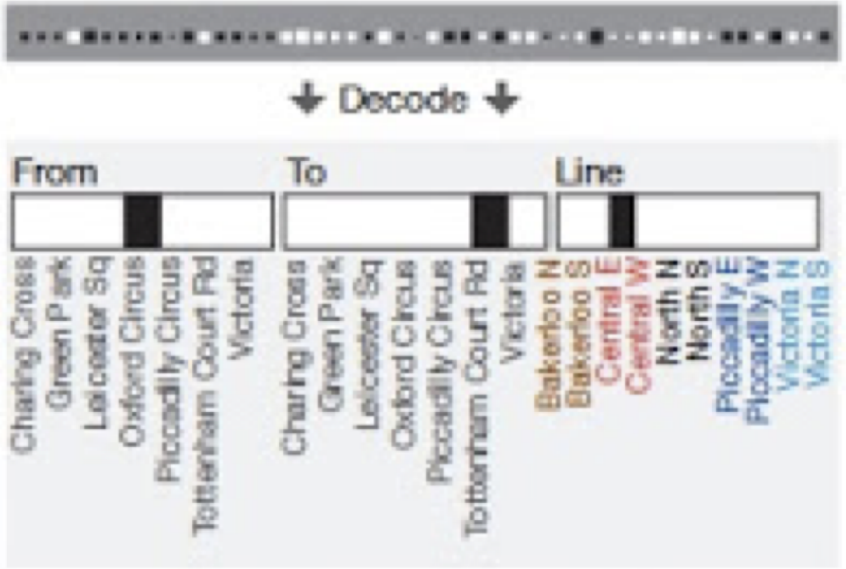
c London Underground map



d Read key



e Location content



人工のニューラルネットワークは、感覚の処理、シーケンスの学習、強化学習には、極めて熟練しているのだが、変数やデータ構造を表現したり、長い時間のスケールでデータを格納する能力には限界を持っている。そうした限界は、ニューラルネットワークが、外部メモリを持たないことによるものだ。

この論文で、我々は、「可微分ニューラル・コンピュータ（DNC: Differentiable Neural Computer）」と呼ばれる機械学習モデルを提案する。このモデルは、従来のコンピュータのランダムアクセスメモリに類似した、外部メモリ行列との間で読み書きが可能なニューラルネットワークで構成される。それは、従来のコンピュータと同じ様に、複雑なデータ構造を表現し操作するためにメモリを使用できるが、ニューラルネットワークのように、データからそれを行う方法を学ぶことができる。

ニューラル・ネットワークの限界

DNCを教師あり学習で訓練すると、それは、自然言語で書かれた論証や推論の問題を真似するように作られた総合的な質問に正しく答えることができることを示す。

我々は、DNCが、ある特定の点の間の最短経路を見つけ出したり、無作為に生成されたグラフで欠けているリンクを推定したりするタスクを学習することが出来ることを示す。さらに、DNCは、これらのタスクを輸送ネットワークや家系図などの特定のグラフに一般化することもできる。

DNCを強化学習で訓練すると、それは、目標の変化を記号列で指定するブロックを移動させるパズルを解くことが出来た。

これらをまとめると、我々の結果は、DNCが、外部読み書きメモリなしのニューラルネットワークでは到達できない、複雑で構造化されたタスクを解決する能力を持っていることを示している。

自然言語で書かれた論証や推論を扱う

現代のコンピュータは、計算とメモリを分離する。計算は、プロセッサによって実行される。プロセッサーは、計算の対象を出し入れするのに、アドレス可能なメモリを利用することができる。

これにより、2つの重要な利点が得られる。
新しい情報を書き込むための拡張可能なストレージの利用と、
メモリの内容を変数として扱う機能である。

変数はアルゴリズムの汎用性にとって非常に重要である。つまり、同じ手続きを、一つのデータだけでなく他のデータで実行するのに、アルゴリズムは、それが読み出すデータのアドレスを変更するだけでいいからである。

現代のコンピュータでのメモリー利用の利点

コンピュータとは対照的に、人工ニューラルネットワークの計算リソースとメモリーのリソースは、ニューラルネットワークの重みとニューロンの活動の中に、混じり合っている。

これは、大きな負債である。タスクの増大によってメモリが必要とされても、これらのネットワークは、新しいストレージを動的に割り当てることもできず、タスクの変数によって実現される、値とは独立に動作するアルゴリズムも簡単には学習できないからである。

ニューラル・ネットワークの弱点

最近のブレークスルーは、ニューラルネットワークは感覚の処理、シーケンスの学習と強化学習には、極めて熟練していることを示しているのだが、認知科学者や神経科学者は、ニューラルネットワークには、変数やデータ構造を表現する能力と外部からの干渉なしに長い期間にわたってデータを格納する能力では限界があると論じてきた。

我々は、ニューラルネットワークに外部メモリへの読み書きのアクセスを提供することで、ニューラルコンピューティングとコンピュータによる処理の利点を組み合わせることを目指している。

メモリーへのアクセスは、記憶間の干渉を最小限に抑える、かつ、長期間の保存を可能にする為に、狭いところに集中して行われる。
システム全体は微分可能であり、それ故、勾配降下法でエンドツーエンドで訓練することができ、ネットワークは目標指向の方法でメモリを操作し、編成する方法を学習できる。

ニューラルネットワークに外部メモリへの読み書きのアクセスを提供する

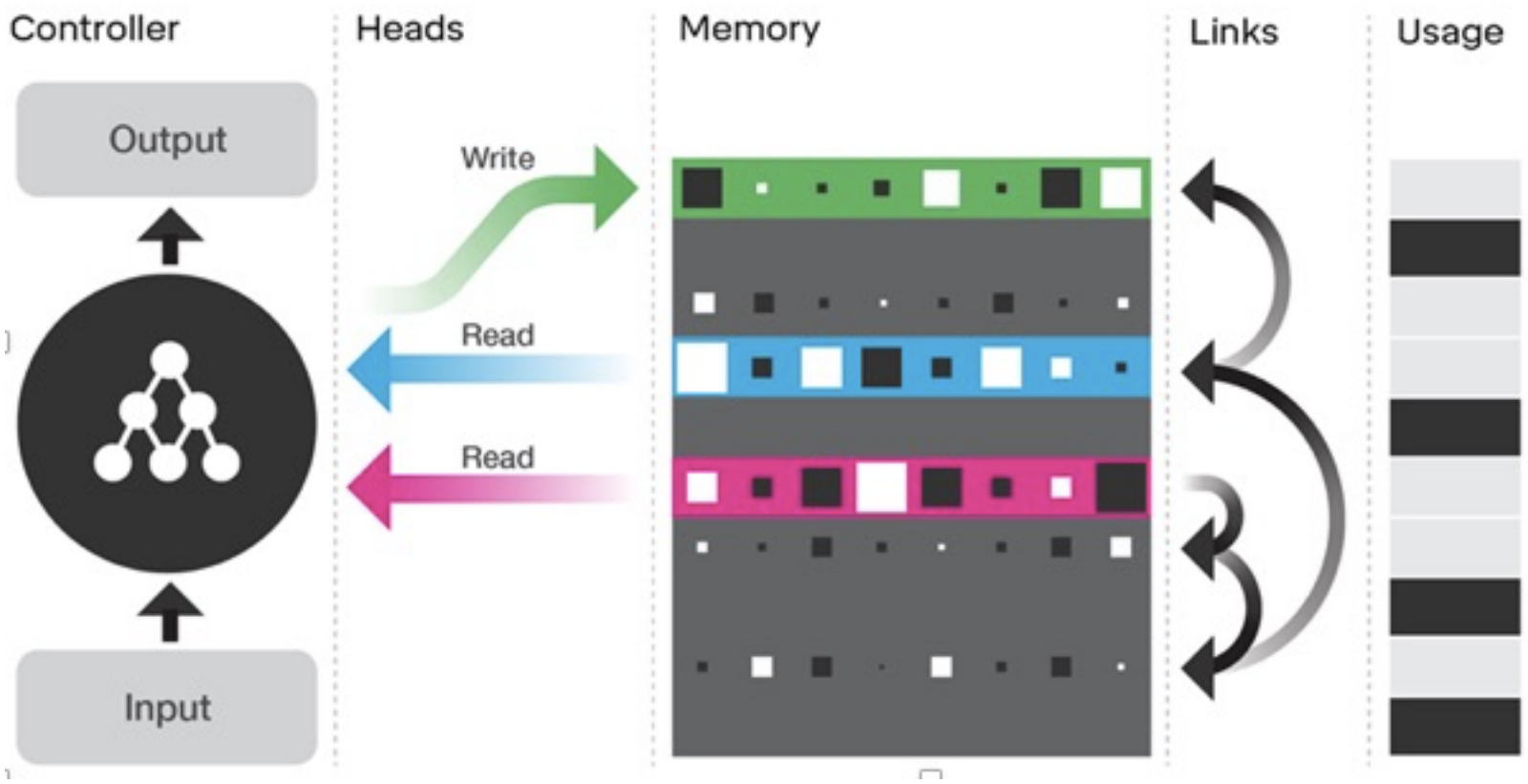
DNCシステムの概観

DNCは、ニューラル・ネットワークをCPUとして、外部メモリー行列をRAMとするコンピュータである

- DNCは外部メモリ行列に結合されたニューラルネットワークである。（ネットワークの動作は、メモリが容量まで満たされていない限りメモリのサイズに依存しないため、メモリは「外部」と見なされる）。このメモリ行列をDNCのRAMと考えるならば、「コントローラ」と呼ばれるネットワークは、微分可能なCPUである。その動作は、勾配降下法で学習される。
- DNCアーキテクチャは、メモリを選択的に読み書きすることができ、メモリの内容を繰り返して修正することが可能である点で、近年のニューラル・メモリー・フレームワークとは異なっている。DNCの以前の形式であるニューラル・チューリング・マシンは、同様の構造を持っていたが、メモリアクセスの方法がより制限されていた。

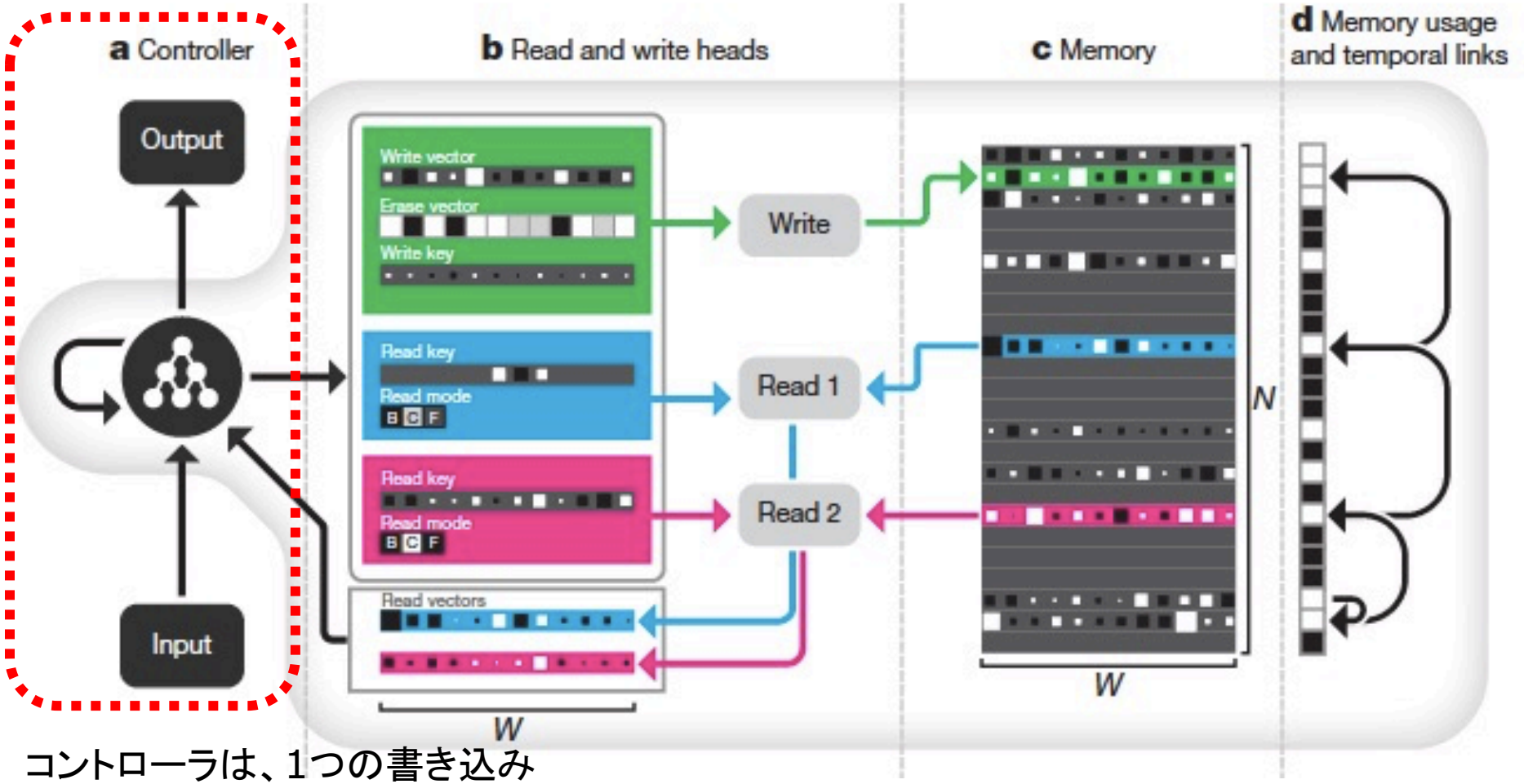
DNC論文から

DNCシステムの概観


Controller
Heads
Memory
Links
Usage
Output
Input
Write
Read
Read

Controller

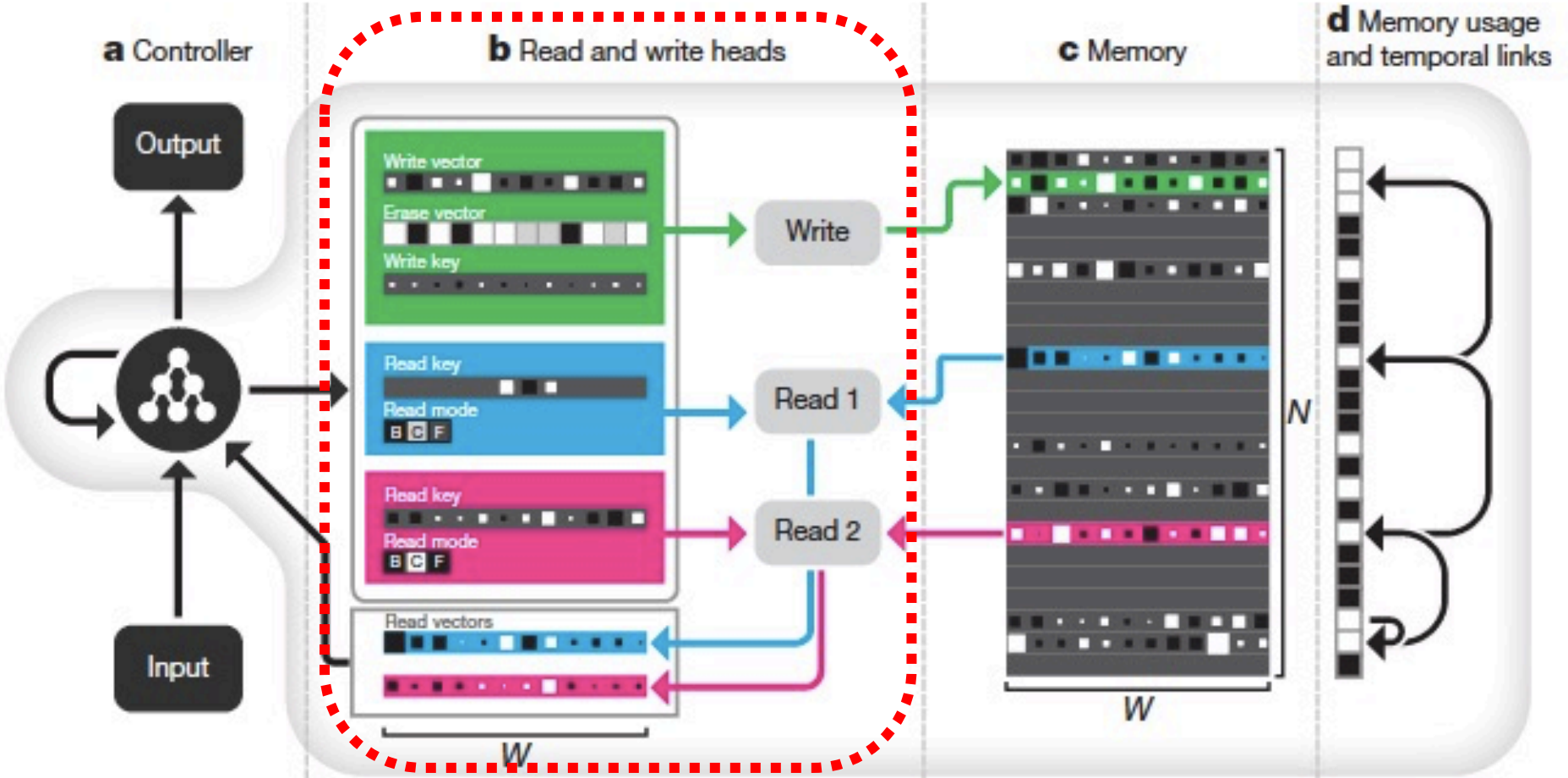
リカレント・コントローラ・ネットワークは、
外部のデータソースから入力を受け取り、
出力を生成する。



コントローラは、1つの書き込み
ヘッド（緑）と複数の読み取りヘッド
（この場合は2つの青とピンク）を
パラメータ化するベクトルも出力する。

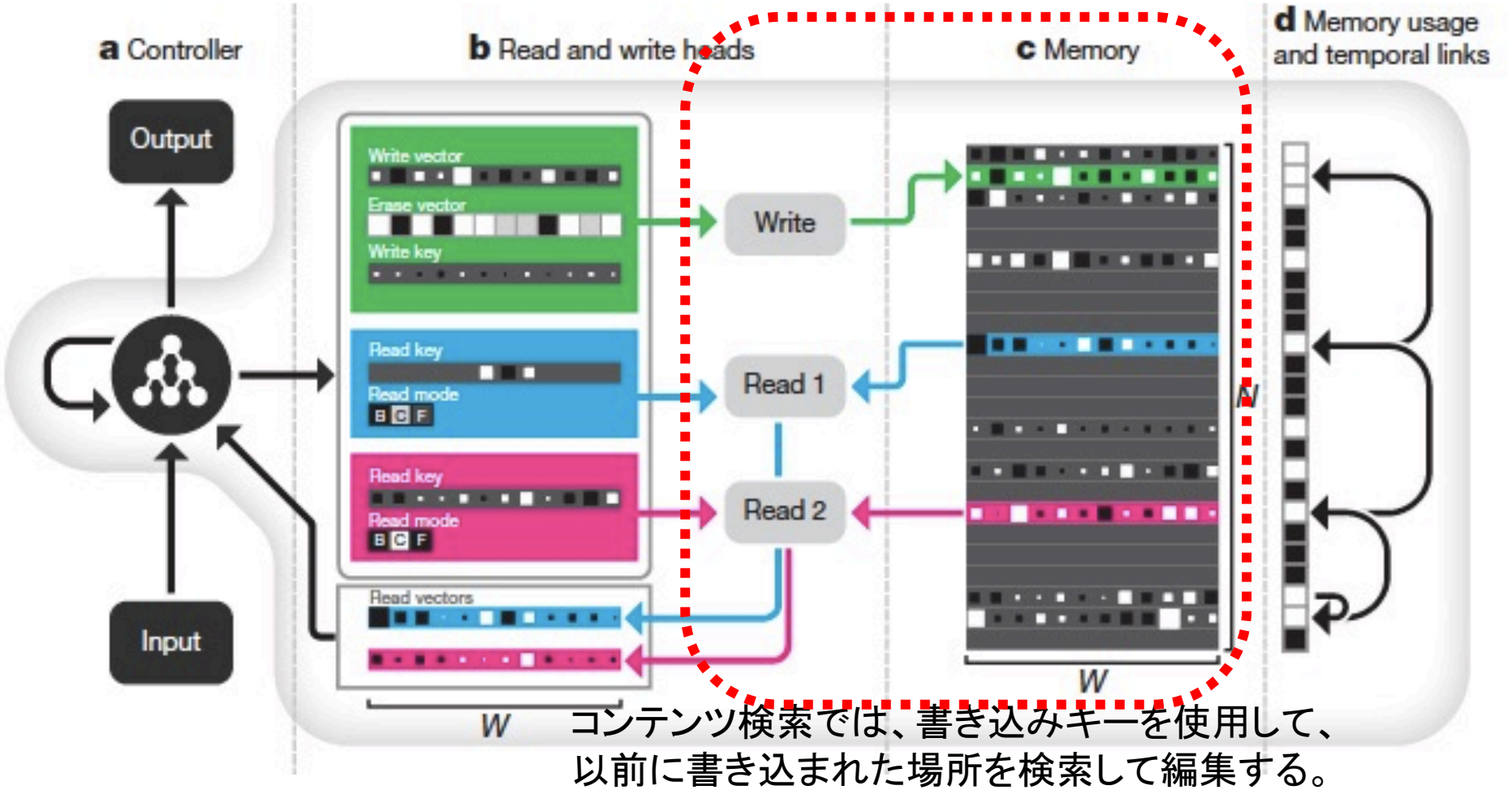
Head

書き込みヘッドは、N × W メモリ行列を編集するために使用される書き込みベクトルおよび消去ベクトルを定義する。



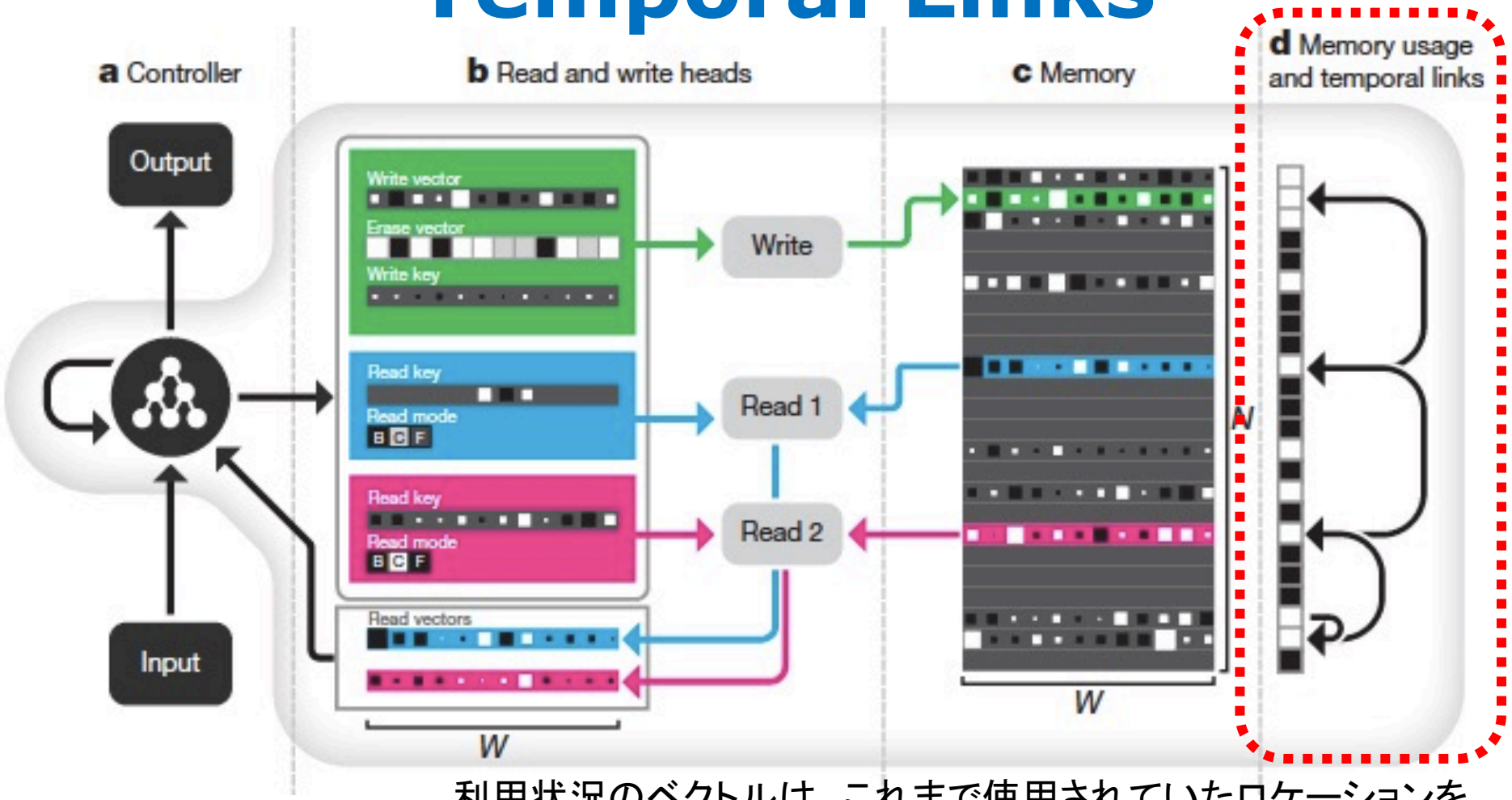
読み取りヘッドは、読み取りモードと呼ばれるゲートを使用して、読み取りキー（ 'C'）を使用したコンテンツ検索と、書かれた順に前方（ 'F'）または後方（ 'B'）へ進む読み出しとを切り替える。

Content Lookup



コンテンツ検索では、書き込みキーを使用して、
以前に書き込まれた場所を検索して編集する。
書込みキーは、メモリ行列内の行 すなわちロケーションに
わたって書込み動作を選択的に集中させる重み付けの
定義に、影響を与えることができる。

Memory Usage and Temporal Links



利用状況のベクトルは、これまで使用されていたロケーションを記録し、時間的リンク行列は、ロケーションに書き込まれた順序を記録する。ここでは、ロケーションの順序は、矢印で表されている。

実験結果をどう見るのか？
baBi task

実験結果をどう見るのか？　baBi task

- baBi task については、”best result” と “mean result” の両方を見た方がいいように思う。「苦手な問題」があるように思う。これは、おそらくFacebookのMemory Network モデルでも同様に見える。
- 結果的には、 ”best result” でも「帰納」と「位置推論」で失敗しているが、根拠から演繹的に推論するTask2, Task3も、 “mean result” を見る限り、安定していないように見える。
- 比較するには、いくつかの前提が必要なのだが、論理的推論能力に関していえば、最も最初期の自動証明プログラムであるSimonの”LOGIC THEORIST” （「プリンキピァ・マテマティカ」の最初の52の定理のうち38の定理を証明できた）の能力を大きく超えてはいないようにも思う。

実験結果をどう見るのか？ baBi task

- 具体的な文が直接に表すものを理解するのと、そこから論理的構造を抽象して、それに基づいて推論するのとは、違うレベルの「知能」の働きなのだ。
- おそらく、そこには超えなければいけない課題が何かある。

bAbI best results

Task	bAbI Best Results						
	LSTM (Joint)	NTM (Joint)	DNC1 (Joint)	DNC2 (Joint)	MemN2N (Joint) [21]	MemN2N (Single) [21]	DMN (Single) [20]
1: 1 supporting fact	24.5	31.5	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**
2: 2 supporting facts	53.2	54.5	1.3	0.4	1.0	**0.3**	1.8
3: 3 supporting facts	48.3	43.9	2.4	**1.8**	6.8	2.1	4.8
4: 2 argument rels.	0.4	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**
5: 3 argument rels.	3.5	0.8	**0.5**	0.8	6.1	0.8	0.7
6: yes/no questions	11.5	17.1	**0.0**	**0.0**	0.1	0.1	**0.0**
7: counting	15.0	17.8	**0.2**	0.6	6.6	2.0	3.1
8: lists/sets	16.5	13.8	**0.1**	0.3	2.7	0.9	3.5
9: simple negation	10.5	16.4	**0.0**	0.2	**0.0**	0.3	**0.0**
10: indefinite knowl.	22.9	16.6	0.2	0.2	0.5	**0.0**	**0.0**
11: basic coreference	6.1	15.2	**0.0**	**0.0**	**0.0**	0.1	0.1
12: conjunction	3.8	8.9	0.1	**0.0**	0.1	**0.0**	**0.0**
13: compound coref.	0.5	7.4	**0.0**	0.1	**0.0**	**0.0**	0.2
14: time reasoning	55.3	24.2	0.3	0.4	**0.0**	0.1	**0.0**
15: basic deduction	44.7	47.0	**0.0**	**0.0**	0.2	**0.0**	**0.0**
16: basic induction	52.6	53.6	52.4	55.1	**0.2**	51.8	0.6
17: positional reas.	39.2	25.5	24.1	**12.0**	41.8	18.6	40.4
18: size reasoning	4.8	2.2	4.0	**0.8**	8.0	5.3	4.7
19: path finding	89.5	4.3	**0.1**	3.9	75.7	2.3	65.5
20: agent motiv.	1.3	1.5	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**
Mean Err. (%)	25.2	20.1	4.3	**3.8**	7.5	4.2	6.4
Failed (err. > 5%)	15	16	**2**	**2**	6	3	**2**

Google

Facebook

bAbI mean results あまり成績良くない

	bAbI Mean Results			
Task	LSTM	NTM	DNC1	DNC2
1: 1 supporting fact	28.4 ± 1.5	40.6 ± 6.7	**9.0 ± 12.6**	16.2 ± 13.7
2: 2 supporting facts	56.0 ± 1.5	56.3 ± 1.5	**39.2 ± 20.5**	47.5 ± 17.3
3: 3 supporting facts	51.3 ± 1.4	47.8 ± 1.7	**39.6 ± 16.4**	44.3 ± 14.5
4: 2 argument rels.	0.8 ± 0.5	0.9 ± 0.7	**0.4 ± 0.7**	**0.4 ± 0.3**
5: 3 argument rels.	3.2 ± 0.5	1.9 ± 0.8	**1.5 ± 1.0**	1.9 ± 0.6
6: yes/no questions	15.2 ± 1.5	18.4 ± 1.6	**6.9 ± 7.5**	11.1 ± 7.1
7: counting	16.4 ± 1.4	19.9 ± 2.5	**9.8 ± 7.0**	15.4 ± 7.1
8: lists/sets	17.7 ± 1.2	18.5 ± 4.9	**5.5 ± 5.9**	10.0 ± 6.6
9: simple negation	15.4 ± 1.5	17.9 ± 2.0	**7.7 ± 8.3**	11.7 ± 7.4
10: indefinite knowl.	28.7 ± 1.7	25.7 ± 7.3	**9.6 ± 11.4**	14.7 ± 10.8
11: basic coreference	12.2 ± 3.5	24.4 ± 7.0	**3.3 ± 5.7**	7.2 ± 8.1
12: conjunction	5.4 ± 0.6	21.9 ± 6.6	**5.0 ± 6.3**	10.1 ± 8.1
13: compound coref.	7.2 ± 2.3	8.2 ± 0.8	**3.1 ± 3.6**	5.5 ± 3.4
14: time reasoning	55.9 ± 1.2	44.9 ± 13.0	**11.0 ± 7.5**	15.0 ± 7.4
15: basic deduction	47.0 ± 1.7	46.5 ± 1.6	**27.2 ± 20.1**	40.2 ± 11.1
16: basic induction	**53.3 ± 1.3**	53.8 ± 1.4	53.6 ± 1.9	54.7 ± 1.3
17: positional reas.	34.8 ± 4.1	**29.9 ± 5.2**	32.4 ± 8.0	30.9 ± 10.1
18: size reasoning	5.0 ± 1.4	4.5 ± 1.3	**4.2 ± 1.8**	4.3 ± 2.1
19: path finding	90.9 ± 1.1	86.5 ± 19.4	**64.6 ± 37.4**	75.8 ± 30.4
20: agent motiv.	1.3 ± 0.4	1.4 ± 0.6	**0.0 ± 0.1**	**0.0 ± 0.0**
Mean Err. (%)	27.3 ± 0.8	28.5 ± 2.9	**16.7 ± 7.6**	20.8 ± 7.1
Failed (err. > 5%)	17.1 ± 1.0	17.3 ± 0.7	**11.2 ± 5.4**	14.0 ± 5.0

bAbi Task 16

Basic induction

1 Lily is a frog.
2 Bernhard is a frog.
3 Bernhard is green.
4 Brian is a lion.
5 Brian is white.
6 Julius is a swan.
7 Julius is green.
8 Lily is green.
9 Greg is a swan.
10 What color is Greg? green 9 6 7

DNCが苦手な推論

bAbi Task 17

Positional reasoning

1 The triangle is above the pink rectangle.
2 The blue square is to the left of the triangle.
3 Is the pink rectangle to the right of the blue square? yes 1 2
4 Is the blue square below the pink rectangle? no 2 1
5 Is the blue square to the right of the pink rectangle? no 2 1
6 Is the blue square below the pink rectangle? no 2 1
7 Is the blue square below the pink rectangle? no 2 1
8 Is the pink rectangle to the left of the blue square? no 1 2

実験結果をどう見るのか？ Graph task

- Graph taskについては、もう一つの「方法論的」と言っていい、悩ましい問題がある。それは、コンピュータは、グラフの問題を別の方法でも解くことができるということである。
- 「最小経路」の問題は、Dijkstraによるよく知られたアルゴリズムがある。スマホの経路検索でも普通に使われているものだ。DNCは、「総当たり」法を発見し、それを試そうとしているように見える。DNCは、Dijkstraの方法を発見できるのだろうか？ 多分、無理だと思う。
- もっと大事なことは、グラフで表現された知識を大規模なシステムで処理するというのは、ディープ・ラーニングでもルール・ベースでもないのだが、広い意味での、そして最も身近な、現代の「人工知能」技術の、すでに根幹であるということだ。
- 2014年2月のマルレク「大規模グラフデータ処理」 https://goo.gl/n8WuRb を参照されたい。

実験結果をどう見るのか？

- Graph Taskのサンプルは、僕には、デジャブだった。
  これは、Prologの練習問題だ。このレベルのデータ量なら、すぐ解けるはずだ。
- Block Taskは、やはり、WinogradのSHRDLUとの比較が必要だと思う。強化学習を使うというのだが、baBi Task 17の、位置の推論が苦手というのは、こうした課題には、大きなハンディだと思う。

DNCはスケールできるか？

- Gravesはいう。
  「実世界のデータに取り組むためには、数千または数百万のロケーションにスケールアップする必要がある。その時点で、メモリはコントローラの重みに含まれる情報より多くの情報を格納できる。」
- ただ、DNCのアーキテクチャーを、そうした規模に拡張できるのだろうか？

- 実際に、DNCが三つの実験に使ったメモリーのロケーションの数は、最大256である。

三つの実験でのDNCのメモリー・サイズ

	bAbI			
	LSTM	NTM	DNC1	DNC2
LSTM Size	512	256	256	256
Batch Size	1	1	1	1
Learning Rate	1×10^{-4}	1×10^{-4}	1×10^{-4}	1×10^{-4}
Memory Dimensions	–	256 × 64	256 × 64	256 × 32
Read Heads	–	4	4	8
Async. Workers	16	16	16	16
DAGGER β	–	–	–	–
λ	–	–	–	–
Entropy Cost Coeff.	–	–	–	–

Extended Data Table 2 | Hyper-parameter settings for bAbI, graph ta

	Graph Tasks			Mini-SHRDLU		
	Shortest Path	Traversal	Inference Tasks	Fig 4 a DNC	Fig 4 a LSTM	Figure 5 DNC
LSTM Size	2 × 256	3 × 256	3 × 256	2 × 250	2 × 250	2 × 250
Batch Size	1	2	32	32	32	32
Learning Rate	3×10^{-6}	1×10^{-5}	1×10^{-5}	3×10^{-5}	3×10^{-5}	3×10^{-5}
Memory Dimensions	128 × 50	256 × 50	128 × 50	32 × 100	–	32 × 100
Read Heads	5	5	5	3	–	2
Async. Workers	–	–	–	–	–	–
DAGGER β	0.8	–	–	–	–	–
λ	–	–	–	0.75	0.5	0.5
Entropy Cost Coeff.	–	–	–	0.5	0.5	0.5

GitHubでのDNCのメモリー・サイズ

```
83
84      def __init__(self,
85                   memory_size=128,
86                   word_size=20,
87                   num_reads=1,
88                   num_writes=1,
89                   name='memory_access'):
90        """Creates a MemoryAccess module.
91
```

https://github.com/deepmind/dnc/blob/master/access.py

DNCはスケールできるか？

- N × N のTemporal Memory Link 行列の計算では、メモリーでも計算時間でも、O(N^2) のリソースが必要になるのだが、上位k個の要素以外は、全てゼロにして、疎な行列を作成すれば、計算時間で*O(NlogN)*のコストで、メモリーで*O(N)*のコストで、無視できない損失なしで近似できるという。（ただし、GitHubでは実装されていない）
- ただ、この手法は、DNCの特徴である、メモリー全域にわたるread/write にも適用できるのだろうか？

bAbI task

http://www.thespermwhale.com/jaseweston/babi/tasks_1-20_v1-2.tar.gz

bAbi Task 2

Two supporting facts

1 Mary got the milk there.

2 John moved to the bedroom.

3 Sandra went back to the kitchen.

4 Mary travelled to the hallway.

5 Where is the milk? hallway 1 4

6 John got the football there.

7 John went to the hallway.

8 Where is the football? hallway 6 7

9 John put down the football.

10 Mary went to the garden.

11 Where is the football? hallway 9 7

12 John went to the kitchen.

13 Sandra travelled to the hallway.

14 Where is the football? hallway 9 7

15 ....

bAbi Task 3

Three supporting facts

1 Mary moved to the bathroom. 2 Sandra journeyed to the bedroom.
3 Mary got the football there.
4 John went back to the bedroom.
5 Mary journeyed to the office.
6 John journeyed to the office.
7 John took the milk.
8 Daniel went back to the kitchen.
9 John moved to the bedroom.
10 Daniel went back to the hallway.
11 Daniel took the apple.
12 John left the milk there.
13 John travelled to the kitchen.
14 Sandra went back to the bathroom.
15 Daniel journeyed to the bathroom.
16 John journeyed to the bathroom.
17 Mary journeyed to the bathroom.
18 Sandra went back to the garden.
19 Sandra went to the office.
20 Daniel went to the garden.
21 Sandra went back to the hallway.
22 Daniel journeyed to the office.
23 Mary dropped the football.
24 John moved to the bedroom.
25 Where was the football before the bathroom? office 23 17 5

bAbi Task 16

Basic induction

1 Lily is a frog.
2 Bernhard is a frog.
3 Bernhard is green.
4 Brian is a lion.
5 Brian is white.
6 Julius is a swan.
7 Julius is green.
8 Lily is green.
9 Greg is a swan.
10 What color is Greg? green 9 6 7
1 Julius is a lion.
2 Lily is a rhino.
3 Bernhard is a swan.
4 Lily is white.
5 Bernhard is green.
6 Greg is a rhino.
7 Greg is gray.
8 Julius is white.
9 Brian is a lion.
10 What color is Brian? white 9 1 8

bAbi Task 17

Positional reasoning

1 The triangle is above the pink rectangle.
2 The blue square is to the left of the triangle.
3 Is the pink rectangle to the right of the blue square? yes 1 2
4 Is the blue square below the pink rectangle? no 2 1
5 Is the blue square to the right of the pink rectangle? no 2 1
6 Is the blue square below the pink rectangle? no 2 1
7 Is the blue square below the pink rectangle? no 2 1
8 Is the pink rectangle to the left of the blue square? no 1 2
9 Is the blue square to the left of the pink rectangle? yes 2 1
10 Is the pink rectangle to the right of the blue square? yes 1 2
1 The red sphere is to the left of the yellow square.
2 The red sphere is below the pink rectangle.
3 Is the pink rectangle to the right of the yellow square? no 2 1
4 Is the yellow square to the right of the pink rectangle? yes 1 2
5 Is the pink rectangle to the left of the yellow square? yes 2 1
6 Is the yellow square below the pink rectangle? yes 1 2
7 Is the yellow square above the pink rectangle? no 1 2
8 Is the yellow square to the right of the pink rectangle? yes 1 2
9 Is the yellow square below the pink rectangle? yes 1 2
10 Is the pink rectangle below the yellow square? no 2 1

bAbi Task 19

Path finding

1 The office is east of the hallway.
2 The kitchen is north of the office.
3 The garden is west of the bedroom.
4 The office is west of the garden.
5 The bathroom is north of the garden.
6 How do you go from the kitchen to the garden? s,e 2 4

1 The bedroom is west of the hallway.
2 The office is east of the garden.
3 The garden is north of the kitchen.
4 The kitchen is north of the bathroom.
5 The hallway is west of the garden.
6 How do you go from the kitchen to the hallway? n,w 3 5

1 The bedroom is south of the hallway.
2 The bathroom is east of the office.
3 The kitchen is west of the garden.
4 The garden is south of the office.
5 The office is south of the bedroom.
6 How do you go from the garden to the bedroom? n,n 4 5

Part II

ボイス・アシスタント・システム
Entity Modelと知識検索



Part II
ボイス・アシスタント・システム
Entity Modelと知識検索　概説

- ボイス・アシスタント・システムの最も基礎的な技術は、2012年にHintonらが開発した、ディープラーニング技術を用いた音声認識技術である。ただ、こうしたSpeech2Text, Text2Speech という機能以外の部分には、ディープラーニング技術は使われていないと思っていい。
- ボイス・アシスタント・システムの最初の商業プロダクトでの成功例は、Googleの音声検索システムである。（2008年）その、バックエンドでは、Entityモデルに基づいたグラフ検索エンジンが走っていた。それはまた、GoogleのKnowledge Graphという新しい検索モデルの導入と連動していた。

Part II
ボイス・アシスタント・システム
Entity Modelと知識検索　概説

- Googleらが設立した、Entity Modelのスキーマを標準として定義しようとする組織 Schema.orgは、Entity Modelの世界では、参照枠として、影響力を持っている。
- ボイスのインターフェースを持たず、限られた目的（クイズ番組Jeopardyで人間と対抗する）の為の試験プロジェクトだったが、IBM Watson（2011年）は、自然言語の理解でも、パーソナル・アシスタント・システムとしても、当時、最高の達成だったと思う。それは、Wikipediaの全てのテキストを「知識」として持ち、システムは、基本的には検索とその結果の評価を高速で行うことで、正解にたどり着く。同時に、プロジェクト内では、検索に限らず、様々な多様なアプローチの模索が行われたことは、特筆に値すると、僕は考えている。

Part II
ボイス・アシスタント・システム
Entity Modelと知識検索　概説

- AmazonのEcho/Alexaは、当初は、発話を束ねるIntent 以外の抽象を持たず、Speech2Text, Text2Speech の能力に依存して、「ああいえば、こういう」というシナリオを、全て人間が用意するという、非常にプリミティブなシステムとして出発する。
- ただ、その後、Intentを補う情報としてのSlotが導入され、あわせて、Slotにマッチする語を事前に準備できるSlot Typeが提供される。また、その後、Intent自身のビルトイン・ライブラリーと、そのインターフェースとして、Intent Signatureが導入される。人工的なシステムで起きていることだが、ある意味で、言語システムの「進化」を見ているようで、興味ふかい。Intent Signatureは、「文法性」への志向としても理解できるのである。

Part II　Agenda

ボイス・アシスタント・システム
Entity Modelと知識検索

- ボイス・アシスタント・システムのプロダクトを見る
- ディープラーニングを用いた音声認識技術 – Hinton
- Google 音声検索 / Google NowとGoogle Assistant
- Google Home
  - DialogFlow
- 検索グラフの構造と Entityモデル
- Amazon Echo / Alexa
  - Intent Model
  - Intent + Slot
  - Slot Typeの導入
  - Intent Signature

ボイス・アシスタント・システムのプロダクトを見る

ここでは、主要なボイス・アシスタントのプロダクトを見ておこう。アプライアンスとして提供されるものが目につくが、スマホ・PCで稼働するもの、SDK/APIを提供するものなど、プロダクトの裾野は広い。

Amazon Alexa



Amazon Alexa Show



Amazon Echo Spot







2017年9月1日、ベルリンで行われたIFA（国際コンシューマー・エレクトロニクス展）で、Google Assistant を搭載した多数の Smart Speakerが発表された。

Sony



Panasonic



JBL



Anker



https://goo.gl/mwzexD




HomePod


Apple-designed
A8 chip
High-excursion
woofer
Six-microphone
array
Seamless mesh
fabric
Seven-tweeter array

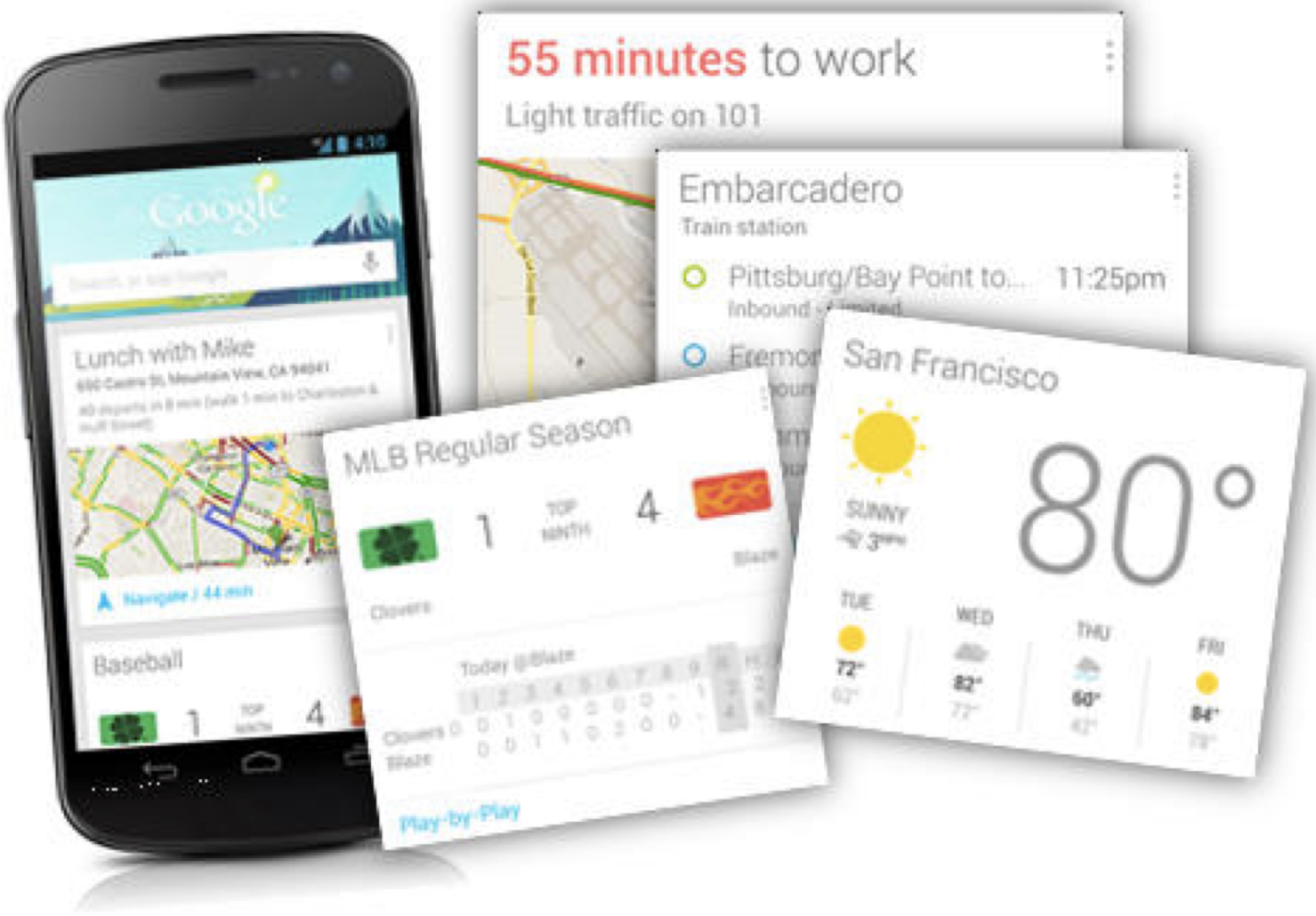


Clova Smart Speaker WAVE

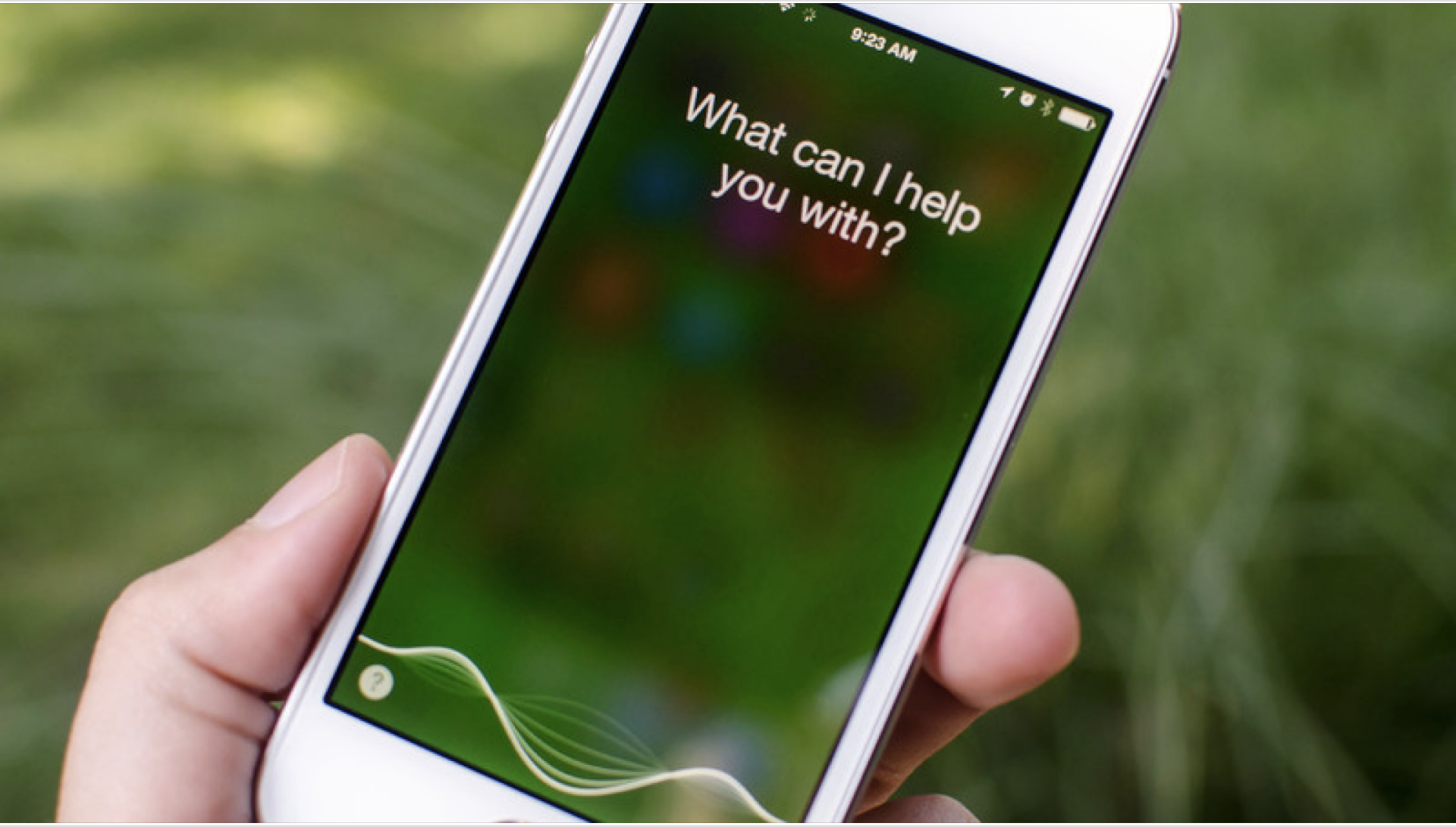
Clova INSIDE



Google Now


55 minutes to work
Light traffic on 101
Embarcadero
Train station
Pittsburg/Bay Point to...
11:25pm
Lunch with Mike
Navigate / 44 min
Baseball
MLB Regular Season
Play-by-play
San Francisco
SUNNY
80°
TUE
WED
THU
FRI

Apple Siri



Facebook Messengers

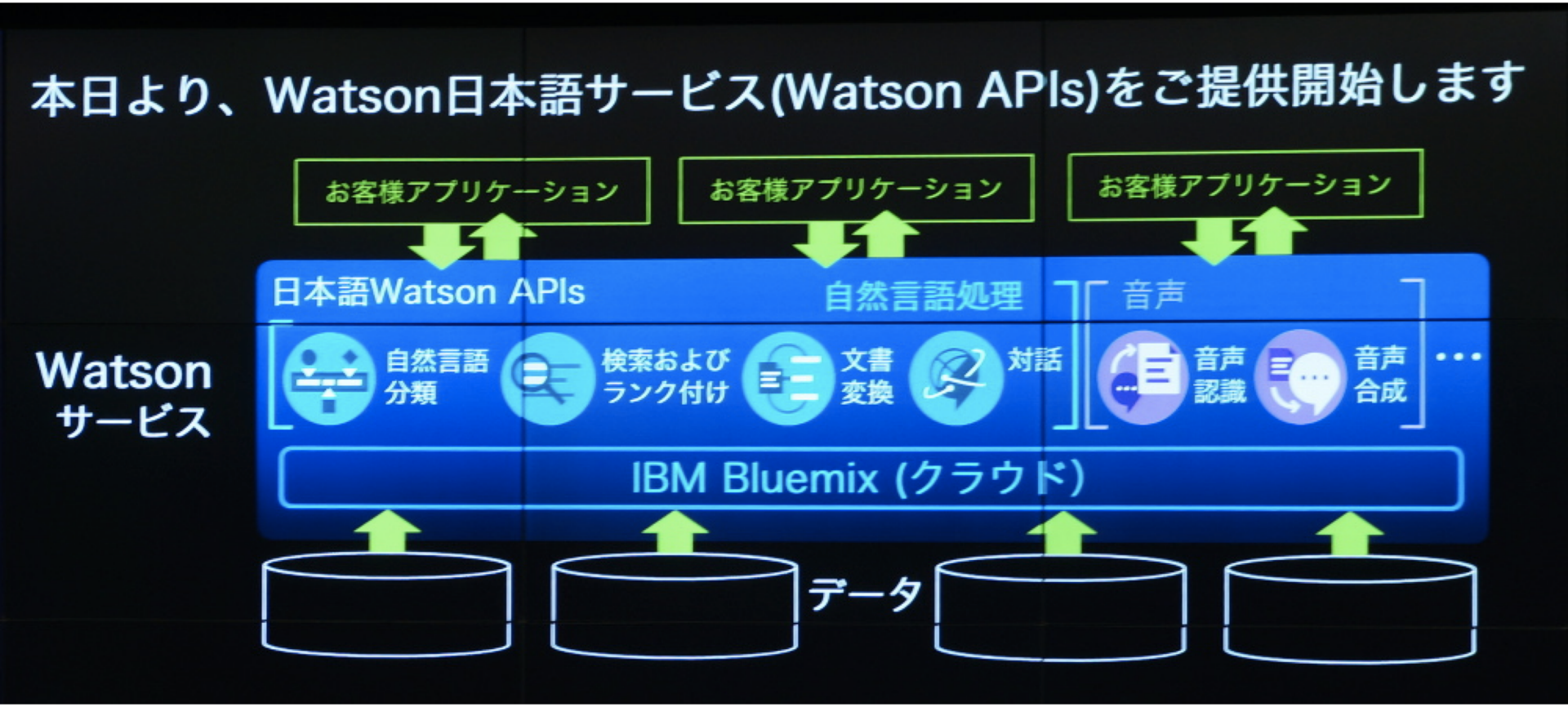


Bots
for Messenger

Microsoft Cortana



IBM Watson API



http://jp.techcrunch.com/2016/02/18/localized-watson-api/

NTT Voice-Series



音声メディア処理技術 を活用した
応用アプリケーション/サービス による 音声ソリューション

Voice-Series (ボイス・シリーズ)

高度で使いやすい音声処理応用技術を開発・提供

NTT研究所による長年の研究開発に裏付けられた音声認識技術や音声合成技術をベースに、
お客様のニーズに添ったソリューションをご提案いたします。

Voice-Series サイト内検索 検索

シリーズラインアップ
Series Line Up

VoiceMall

IVR ソリューション

高機能なクラウド型音声（自動応答）サービス
お客様ニーズの高い業務をパッケージ化

SpeechRec

音声認識ソリューション

高精度な音声認識エンジン
さまざまな用途への適用が可能

FutureVoice

音声合成ソリューション

業界最高クラスの高品質音声合成
話者と口調を自由に組み合わせできる音声合成

VoiceMall **SpeechRec** **FutureVoice**

ディープラーニングを用いた音声認識技術の飛躍

"Deep Neural Networks for Acoustic Modeling in Speech Recognition"

Geoffrey Hinton et al.

https://goo.gl/GT6Bg9

2012年



近年の音声認識システムの多くは、時間変動性を扱うために隠れマルコフモデル（HMM）を用い、各HMMの各状態がフレームまたは音声入力を表す係数のフレームのショートウィンドウにどれだけ適しているかを決定するためガウシアン混合モデルを用いている。

適合性を評価するもう一つの別の方法は、係数のいくつかのフレームを入力として受け取り、HMMの状態の事後確率を出力として生成するフィードフォワードのニューラルネットワークを用いることである。多くの隠れ層を持つディープ・ニューラルネットワークは、新しい方法で訓練され、さまざまな音声認識ベンチマークでガウシアン混合モデルより、大幅に優れていることが示されている。

この論文では、この進捗の概要を説明し、音声認識での音響モデルにディープ・ニューラルネットワークを使用することで近年成功している4つの研究グループ（Tront大+Microsoft+Google+IBM ）の共有された見解を述べる。

従来の方法との比較

DNN = Deep Neural Network　デープラーニング
GMM = Gausian Mix Model　　従来の方法
HMM = Hidden Markov Model

task	hours of training data	DNN-HMM	GMM-HMM with same data	GMM-HMM with more data
Switchboard (test set 1)	309	18.5	27.4	18.6 (2000 hrs)
Switchboard (test set 2)	309	16.1	23.6	17.1 (2000 hrs)
English Broadcast News	50	17.5	18.8	
Bing Voice Search (Sentence error rates)	24	30.4	36.2	
Google Voice Input	5,870	12.3		16.0 (>>5,870hrs)
Youtube	1,400	47.6	52.3	

Google音声検索

ボイス・アシスタントが最初に登場したのは、Google の音声検索としてである。

日本語とGoogle音声検索

- Googleが、音声インターフェース機能を備えた音声検索技術をリリースしたのは2008年である。それは英語を対象としたものだった。日本でのGoogle音声検索のリリースは、2009年で、世界で二番目に早かった。
- (音声→文字列)→検索→(検索された文字列→音声) と言う処理を日本語で行う上で、大きな問題が二つあった。
- 一つは、日本語の文に使われている文字種が、ひらがな・カタカナ・漢字・ASCII英字と多様であること。
  もう一つの問題は、欧米語の文が、基本的にはスペースで区切られて語の境界が明確なのに対して、日本語ではそうした区切りが存在しないこと。（中国語・ハングルでも同様である）
- 発話から、日本語文字列に変換するのが困難な例として、次のようなものがあった。「AKB48」「シャ乱Q」「価格.com」

日本語とGoogle音声検索

- 2008年、Googleは、日本で、すべての年齢・方言・地域をカバーした、500人の日本語の話し手を有給で雇って、数週間で250k種類の発話集を作った。データとしては、その地域で最も多く検索されたトップN個の検索文を使い、それを読み上げさせた。iPhoneとBlackberryが使われたと言う。
- この日本語発話コーパスをベースに、Googleの日本語音声認識の精度は大きく上がった。
- このプロジェクトの最大の成果は、スペースの区切りなしでも、かつ、字句の文法構造の解析抜きでも、文を語に分割する、新しい技術 "WordPiece"が生み出されたことである。
- WordPieceは、後のGoogleニューラル翻訳で、基本的なツールとして活用されることになる。日本語の難しさが、Googleの自然言語処理技術の発展に貢献したのである。

2012年 [35] Schuster, M., and Nakajima, K. ”**Japanese and Korean voice search**” https://goo.gl/DUxQuv

1. まず、単位語の目録を、ベーシックUnicode（日本語の漢字・ひらがな・カタカナ、韓国語のハングル、そしてすべてのASCIIコードを含む）で作る。最終的には、日本語でトータル22000語、韓国語11000の目録になる。
2. この目録を使った訓練データ上に、言語モデルを構築する。
3. 現在の語の目録から二つの単位語を取り出し、一つに結合して新しい単位語を作り、単位語の目録を一つ増やす。可能な組み合わせの中から、言語モデルに追加された時、訓練データ上で、類似度を一番増大させるものを、新しい単位語として選ぶ。
4. 事前に定義された単位語の限界に到達するか、類似度の増大が、ある閾値以下に下がるまで、2. の操作を繰り返す。

Google Cloud Speech API Language Support

□119ヶ国語のサポート

Thai (Thailand)
Korean (South Korea)
Chinese, Mandarin (Traditional, Taiwan)
Chinese, Cantonese (Traditional, Hong Kong)
Japanese (Japan)
Chinese, Mandarin (Simplified, Hong Kong)
Chinese, Mandarin (Simplified, China)

Afrikaans (South Africa)	English (Ghana)	Spanish (Colombia)	Spanish (Dominican Republic)
Amharic (Ethiopia)	English (United Kingdom)	Spanish (Costa Rica)	Spanish (Uruguay)
Armenian (Armenia)	English (India)	Spanish (Ecuador)	Spanish (Venezuela)
Azerbaijani (Azerbaijan)	English (Ireland)	Spanish (El Salvador)	Basque (Spain)
Indonesian (Indonesia)	English (Kenya)	Spanish (Spain)	Filipino (Philippines)
Malay (Malaysia)	English (New Zealand)	Spanish (United States)	French (Canada)
Bengali (Bangladesh)	English (Nigeria)	Spanish (Guatemala)	French (France)
Bengali (India)	English (Philippines)	Spanish (Honduras)	Galician (Spain)
Catalan (Spain)	English (South Africa)	Spanish (Mexico)	Georgian (Georgia)
Czech (Czech Republic)	English (Tanzania)	Spanish (Nicaragua)	Gujarati (India)
Danish (Denmark)	English (United States)	Spanish (Panama)	Croatian (Croatia)
German (Germany)	Spanish (Argentina)	Spanish (Paraguay)	Zulu (South Africa)
English (Australia)	Spanish (Bolivia)	Spanish (Peru)	Icelandic (Iceland)
English (Canada)	Spanish (Chile)	Spanish (Puerto Rico)	Italian (Italy)

Javanese (Indonesia)	Portuguese (Portugal)	Telugu (India)	Arabic (Bahrain)
Kannada (India)	Romanian (Romania)	Vietnamese (Vietnam)	Arabic (Algeria)
Khmer (Cambodia)	Sinhala (Srilanka)	Turkish (Turkey)	Arabic (Saudi Arabia)
Lao (Laos)	Slovak (Slovakia)	Urdu (Pakistan)	Arabic (Iraq)
Latvian (Latvia)	Slovenian (Slovenia)	Urdu (India)	Arabic (Kuwait)
Lithuanian (Lithuania)	Sundanese (Indonesia)	Greek (Greece)	Arabic (Morocco)
Hungarian (Hungary)	Swahili (Tanzania)	Bulgarian (Bulgaria)	Arabic (Tunisia)
Malayalam (India)	Swahili (Kenya)	Russian (Russia)	Arabic (Oman)
Marathi (India)	Finnish (Finland)	Serbian (Serbia)	Arabic (State of Palestine)
Dutch (Netherlands)	Swedish (Sweden)	Ukrainian (Ukraine)	Arabic (Qatar)
Nepali (Nepal)	Tamil (India)	Hebrew (Israel)	Arabic (Lebanon)
Norwegian Bokmål (Norway)	Tamil (Singapore)	Arabic (Israel)	Arabic (Egypt)
Polish (Poland)	Tamil (Sri Lanka)	Arabic (Jordan)	Persian (Iran)
Portuguese (Brazil)	Tamil (Malaysia)	Arabic (United Arab Emira	Hindi (India)

Google Nowにできること

Google AssistantとGoogle Now

Google AssistantとGoogle Now

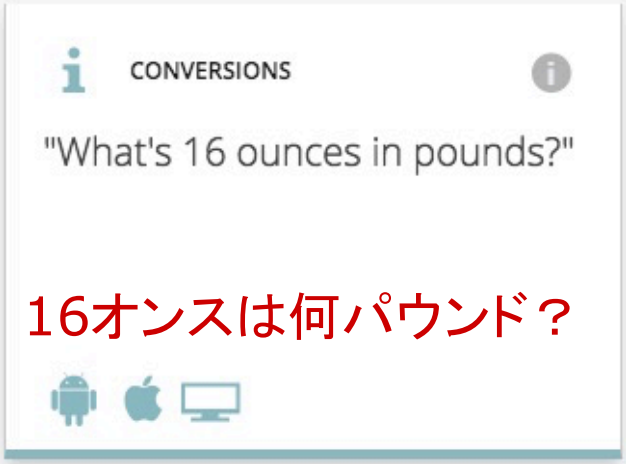
- ボイス・アシスタントに先行したのは、音声検索である。その最大の成功例は、GoogleのGoogle Nowである。
- ボイス・アシスタントとしてのGoogle Assistantと、音声検索のGoogle Nowは、ほとんど、同じものである。
- Google Nowにできることを見てみよう。ここでは、外部の情報の検索（もちろん、Googleの検索機能である）が必要なものを赤字で、システム内部で実行可能なものを青字で示した。
- ボイス・アシスタントに理解できる文は、疑問文と命令文だけである。こうした「意味のプラグマティックな理解」は、意味の理解を実装するのに、疑問文と命令文が容易だからということに尽きる。

Google Nowができること

外部の情報の検索が必要

システム内部で実行可能

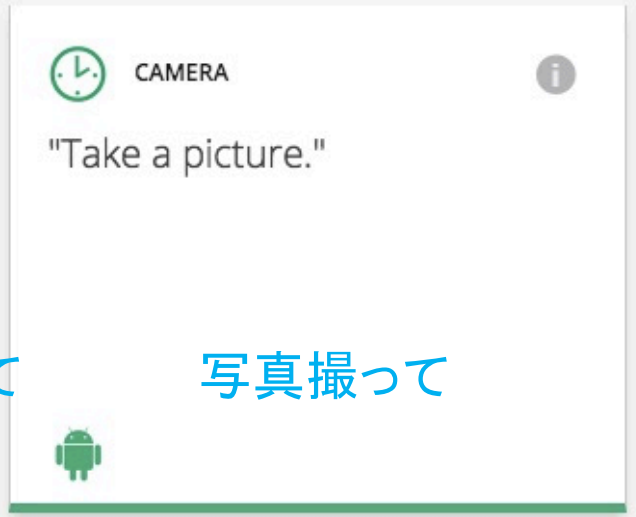












Google Nowは、ボイス・アシスタント・システムである

Google Nowができること

外部の情報の検索が必要

システム内部で実行可能

LOCAL MOVIES

"What movies are playing tonight?"
"Where is Thor playing?"

今日の映画は？

EMAILS

"Email Jamie that the meeting was moved to 3."

予定が変わったとメールして

WEBSITES

"Go to wikipedia.com."

Wikipedia表示して

ENTERTAINMENT TRIVIA

"Who plays Carrie Mattheson on Homeland?"
"Who directed Titanic?"

Titanicの監督は？

FINANCE

"What's Google's stock price?"
"What is Twitter trading at?"

Googleの株価は？

CALENDAR EVENTS

"Create an event, dinner with Jamie at 8 PM."

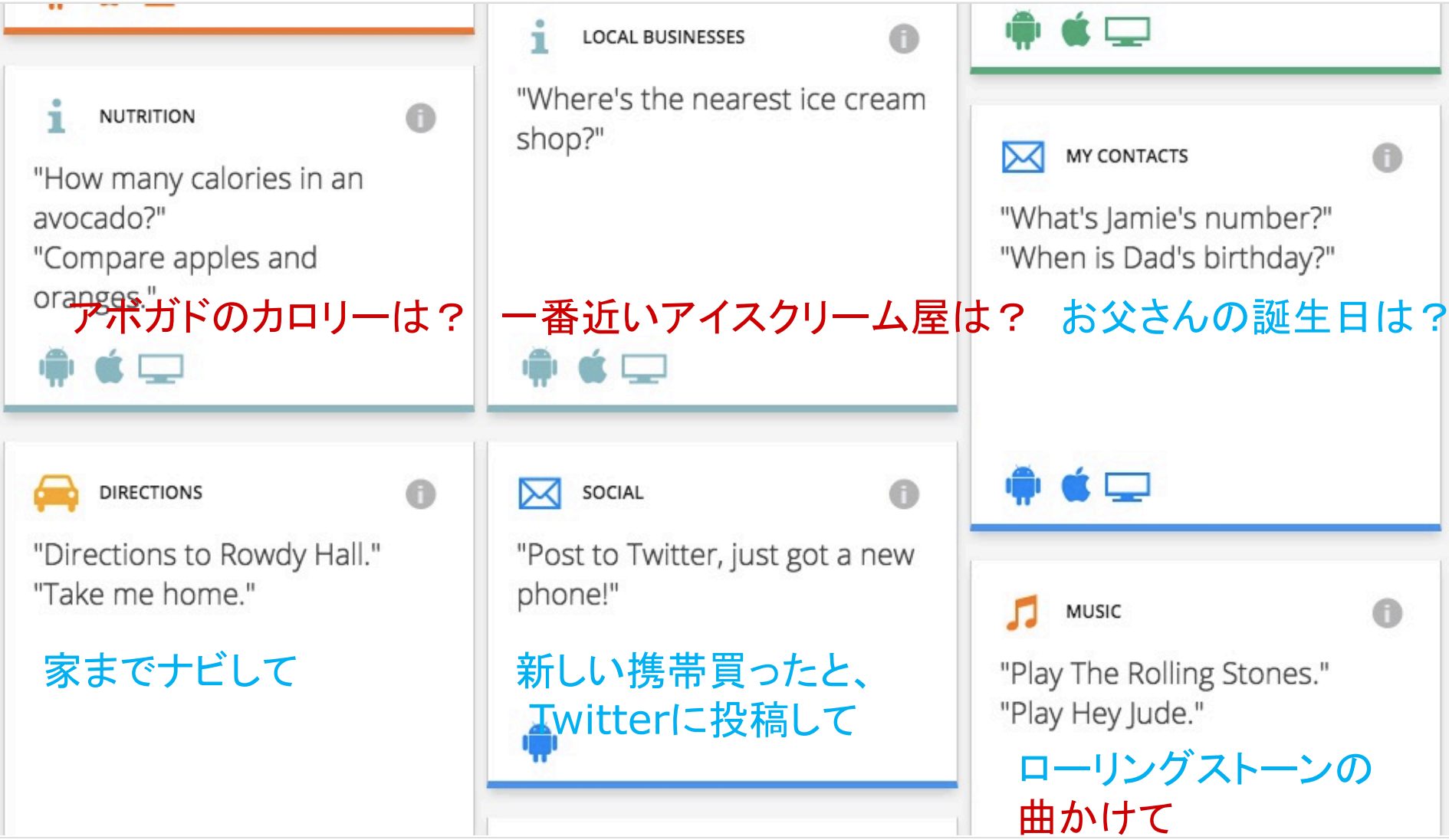
8時にディナーの予定入れて

Google Nowは、ボイス・アシスタント・システムである

Google Nowができること

外部の情報の検索が必要

システム内部で実行可能



Google Nowは、ボイス・アシスタント・システムである

Google Nowができること

外部の情報の検索が必要

システム内部で実行可能

TV SHOWS

"Listen to my TV."

テレビをつけて

FAMOUS PEOPLE

"Where was Albert Einstein born?"
"How old is Brad Pitt?"

アインシュタインは
どこで生まれたの？

NAVIGATION

"Navigate to St. Stephen's Green."

●●までナビして

TRAVEL INFO

"What are some attractions in New York City?"

ニューヨークで、なにか
イベントないかしら？

FAMOUS PEOPLE

"Where was Albert Einstein born?"
"How old is Brad Pitt?"

ブラッド・ピットは何歳？

DICTIONARY

"What does onomatopoeia mean?"

onomatopoelaって
どんな意味？

Google Nowは、ボイス・アシスタント・システムである

二つのタイプのデモ

- 「アインシュタインの誕生日は？」
- 「ボルツマンの死んだ日は？」
- 「ボルツマンの論争相手は？」
- 「Taylor Swiftの誕生日は？」
- 「富士山の高さは？」
- 「稚内市の人口は？」
- 「バッハのマタイ受難曲かけて！」
- 「マーラーの曲かけて！」
- 「ビートルズの音楽かけて！」
- 「丸山にメールを出して！」
- 「丸山に電話して！」
- 「カメラ起動して！」

Google Nowの検索の特徴

- 一般には、Googleの検索は、与えられた文字列を、Googleが収集した膨大なデータから探し出し、検索結果を独自のアルゴリズムで順序づけて、その文書のURLを返すものである。
- ただ、Google Nowで行われている検索は、それとは異なっている。それは、後で見るような **Entity / Property モデル**に基づいて構築された巨大なグラフデータを検索して、マッチしたEntityのPropertyの値を返す。
- それをGoogleは、**Knowledge Graph**と呼んでいる。この検索スタイルの変化は、2010年代に行われた。

Google Nowの検索の問題

- Google NowのKnowkedge Graph検索の問題は、それが、必ずしも網羅的ではないことだ。従来の文字列検索の方が、圧倒的にデータ量が大きい。限られたEntityしか、グラフ上に存在していない。Googleの音声検索では、グラフ上にデータがないと、音声での読み上げを断念し、従来の文字列検索の結果を表示する。
- 先の「ボルツマンの論争相手」では、ボルツマンというEntityは存在するのだが(「誕生日」は答えられた)、そのEntityには、「論争相手」というPropertyが定義されていないことを示している。
- Knowkedge Graph検索の大きな問題は、それにアクセスするAPIが公開されているのだが、その機能が大きく制限されていること。Taylor SwiftのEntityには誕生日が登録されているのだが、公開されているAPIからはアクセスできない。

Google Knowledge Graph Search API

https://developers.google.com/knowledge-graph/

KnowledgeGraphのEntityにアクセスできるが、その機能は、限られている。

Google Knowledge Graph Search API

- The Knowledge Graph Search API lets you find entities in the Google Knowledge Graph. The API uses standard schema.org types and is compliant with the JSON-LD specification.

- “Taylor Swift” についての検索サンプル

```
https://kgsearch.googleapis.com/v1/entities:search?query=taylor+swift&key=API_KEY&limit=1&indent=True
```

```
{
 "@context": {
   "@vocab": "http://schema.org/",
   "goog": "http://schema.googleapis.com/",
   "resultScore": "goog:resultScore",
   "detailedDescription": "goog:detailedDescription",
   "EntitySearchResult": "goog:EntitySearchResult",
   "kg": "http://g.co/kg"
 },
 "@type": "ItemList",
 "itemListElement": [
   {
     "@type": "EntitySearchResult",
     "result": {
       "@id": "kg:/m/0dl567",
       "name": "Taylor Swift",
       "@type": [
         "Thing",
         "Person"
       ],
       "description": "Singer-songwriter",
```

```
"image": {
        "contentUrl":
```



```
      "image": {
        "contentUrl":
"https://t1.gstatic.com/images?q=tbn:ANd9GcQmVDAhjhWnN2OWys2ZMO3PGAhupp5tN2LwF_BJmiHgi19hf8Ku",
        "url": "https://en.wikipedia.org/wiki/Taylor_Swift",
        "license": "http://creativecommons.org/licenses/by-sa/2.0"
      },
      "detailedDescription": {
        "articleBody": "Taylor Alison Swift is an American singer-songwriter and actress. Raised in Wyomissing, Pennsylvania, she moved to Nashville, Tennessee, at the age of 14 to pursue a career in country music. ",
        "url": "http://en.wikipedia.org/wiki/Taylor_Swift",
        "license":
"https://en.wikipedia.org/wiki/Wikipedia:Text_of_Creative_Commons_Attribution-ShareAlike_3.0_Unported_License"
      },
      "url": "http://taylorswift.com/"
    },
    "resultScore": 896.576599
  }
 ]
}
```

Sample Code in Python

```
"""Example of Python client calling Knowledge Graph Search API."""
import json
import urllib

api_key = open('.api_key').read()
query = 'Taylor Swift'
service_url = 'https://kgsearch.googleapis.com/v1/entities:search'
params = {
    'query': query,
    'limit': 10,
    'indent': True,
    'key': api_key,
}
url = service_url + '?' + urllib.urlencode(params)
response = json.loads(urllib.urlopen(url).read())
for element in response['itemListElement']:
  print element['result']['name'] + ' (' + str(element['resultScore']) + ')'
```

Sample Code in JavaScript

```
<!DOCTYPE html>
<html>
<head>
    <script
src="http://ajax.googleapis.com/ajax/libs/jquery/2.1.4/jquery.min.js"></s
cript>
</head>
<body>
<script>
  var service_url = 'https://kgsearch.googleapis.com/v1/entities:search';
  var params = {
    'query': 'Taylor Swift',
    'limit': 10,
    'indent': true,
    'key' : '<put your api_key here>',
  };
  $.getJSON(service_url + '?callback=?', params, function(response) {
    $.each(response.itemListElement, function(i, element) {
      $('<div>',
{text:element['result']['name']}).appendTo(document.body);
    });
  });
</script>
</body>
</html>
```

Google Assistant SDK

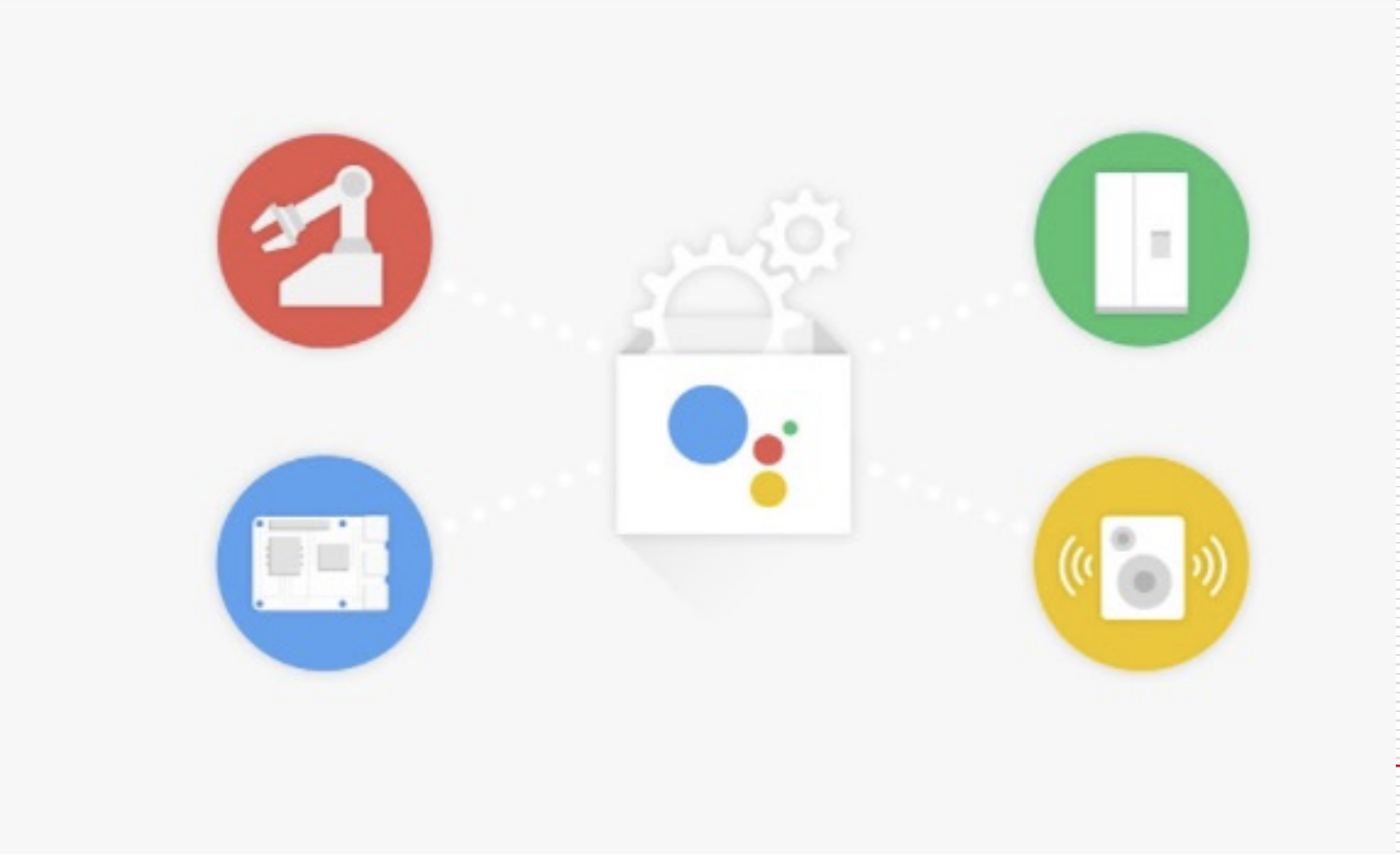
The developer preview (0.0.3) **2017/7/19**

https://developers.google.com/assistant/sdk/

- Google Assistant library for Python
- Google Assistant gRPC API

ここから、開発者のForumに入れる
https://groups.google.com/forum/#!forum/google-assistant-sdk-updates

Embed the Google Assistant in any device



	Library	gRPC API
Supported architectures	linux-armv7l and linux-x86_64	gRPC platforms
Supported languages	Python	gRPC languages
Hands-free activation (*Ok Google*)	Yes	No
Audio capture and playback	Built in	Reference code is provided
Conversation state management	Built in	Reference code is provided
Timers and alarms	Yes	No

google.assistant.embedded.v1alpha2

EmbeddedAssistant

Service that implements the Google Assistant API.

Assist

```
rpc Assist(AssistRequest) returns (AssistResponse)
```

Initiates or continues a conversation with the embedded Assistant Service. Each call performs one round-trip, sending an audio request to the service and receiving the audio response. Uses bidirectional streaming to receive results, such as the **END_OF_UTTERANCE** event, while sending audio.

Converse → **Assist**

Request

Add to my shopping list

Responsee

What do you want to add?.

A conversation is one or more gRPC connections, each consisting of several streamed requests and responses. For example, the user says *Add to my shopping list* and the Assistant responds *What do you want to add?*. The sequence of streamed requests and responses in the first gRPC message could be:

- AssistRequest.config
- AssistRequest.audio_in
- AssistRequest.audio_in
- AssistRequest.audio_in
- AssistRequest.audio_in

オーディオ・データを送ると

"Add to my shopping list"

- AssistResponse.event_type.END_OF_UTTERANCE
- AssistResponse.speech_results.transcript "add to my shopping list"
- AssistResponse.dialog_state_out.microphone_mode.DIALOG_FOLLOW_ON
- AssistResponse.audio_out
- AssistResponse.audio_out
- AssistResponse.audio_out

"What do you want to add?."

オーディオ・データが返る

google.assistant.embedded.v1alpha2

google.assistant SDKの「文盲」プロトコル

Request

bagels

Responsee

OK, I've added bagels to your shopping list.

The user then says *bagels* and the Assistant responds *OK, I've added bagels to your shopping list*. This is sent as another gRPC connection call to the **`Assist`** method, again with streamed requests and responses, such as:

- AssistRequest.config
- AssistRequest.audio_in
- AssistRequest.audio_in
- AssistRequest.audio_in
- AssistResponse.event_type.END_OF_UTTERANCE
- AssistResponse.dialog_state_out.microphone_mode.CLOSE_MICROPHONE
- AssistResponse.audio_out
- AssistResponse.audio_out
- AssistResponse.audio_out
- AssistResponse.audio_out

オーディオ・データを送ると

bagels

OK, I've added bagels to your shopping list.

オーディオ・データが返る

google.assistant.embedded.v1alpha2

google.assistant SDKの「文盲」プロトコル

Introducing Dialogflow, the new name for API.AI

2017/10/10

https://blog.dialogflow.com/post/apiai-new-name-dialogflow-new-features/





- **In-line code editor:** you can now write fulfillment logic, test, and implement a functional webhook directly in the console.
- **Multi-lingual agent support:** building for multiple languages is now easier than ever. You can now add additional languages and locales to your new or existing agent.

Fulfillmentを構成する

- Define your conversation
  - Create intents
- Building responses in Dialogflow
- Building fulfillment responses

Define your conversation
Intentsの生成



Actionは、どのデータがFulfillmentに送られるかを定義する。
このデータには、ユーザーの入力からパーズされたデータや、Intentの名前などが含まれている。



Define your Response



Build Fulfillment



Fulfillmentのコード・サンプル

```
const app = new DialogflowApp({request: request, response: response});

const WELCOME_INTENT = 'input.welcome';  // the action name from the Dialogflow intent
const NUMBER_INTENT = 'input.number';  // the action name from the Dialogflow intent
const NUMBER_ARGUMENT = 'input.mynum'; // the action name from the Dialogflow intent

function welcomeIntent (app) {
  app.ask('Welcome to number echo! Say a number.');
}

function numberIntent (app) {
  let number = app.getArgument(NUMBER_ARGUMENT);
  app.tell('You said ' + number);
}
```

Fulfillmentのコード・サンプル

```
function responseHandler (app) {
  // intent contains the name of the intent you defined in the Actions area of Dialogflow
  let intent = app.getIntent();
  switch (intent) {
    case WELCOME_INTENT:
      app.ask('Welcome! Say a number.');
      break;

    case NUMBER_INTENT:
      let number = app.getArgument(NUMBER_ARGUMENT);
      app.tell('You said ' + number);
      break;
  }
}
```

検索グラフの構造と Entityモデル

検索グラフの構造とボキャブラリーは、スキーマによって定義されている。それをEntityモデルと呼ぶ。

ここでは、GoogleのKnowledge Graphで採用されている、Schema.org のスキーマの特徴を見てみよう。

知識のグラフでの表現と検索

- Knowledge Graph は、”people”, “places”, “things” といった実世界のエンティティを記述する数百万のエンティティを持っている。これらのエンティティが、このグラフのノードを形成する。
- これらのエンティティの「ノード」と、ノード間の関係あるいはアクションを表す「エッジ」のボキャブラリーは、次に見る組織 Schema.orgによって、与えられている。

知識のグラフでの表現と検索

- 例えば、「アインシュタインの誕生日」を知ろうとすれば、以前の文字列ベースの検索では、「アインシュタイン」を検索して、検索エンジンが返す膨大な数の（PageRankで順序付けされているのだが）文字列「アインシュタイン」を含むURLのドキュメントを、検索者が確認して、その中で彼の誕生日の情報を見つける必要がある。
- グラフ・ベースの検索では、Person Entityノードであるアインシュタインの Property **birthdate** の値をしらべれば、それはすぐにわかる。
- 同様に、Person Entityのもつ Propertyである、award, colleague, jobTitle, worksFor などの値を取得すれば、もっと多くの情報が簡単に得られる。

知識のグラフでの表現と検索

- Google Nowの場合なら、イベントの予約先から送られたメールの中から、予約番号、予約者名、イベントの日時・場所等の情報（ただし、Event EntityのPropertyに含まれるものに限る）を、得ることが出来る。
- FacebookのGraph Searchなら、ソーシャル・グラフのノードをたどることで、「サンフランシスコに住んでいる私の友達は誰？」といった質問に答えることができる。

Welcome to Schema.org

http://schema.org/
https://github.com/schemaorg/schemaorg

Schema.orgのスキーマは、Google, Microsoft, Pinterest, Yandex等で使われている。Facebookは、そうではない。

Welcome to Schema.org

- Schema.org は、インターネットやWebページ、e-メールのメッセージやさらにもっと多くのものの上にある構造化されたデータに対して、スキーマを作りあげ、維持し、それを広めるというミッションを持った共同のコミュニティ活動である。
- Schema.org は、**Google, Microsoft, Yahoo, Yandex** のスポンサー支援を得ている。そのボキャブラリーは、メーリングリスト public-schemaorg @w3.org と GitHub https://goo.gl/WvCyP3 を用いて、オープンなプロセスで開発されている。

folksonomyアプローチ

http://schema.org/

Schema.org vocabulary

- Schema.org ボキャクラリーは、RDFa, Microdata, JSON-LD を含む多くのエンコーディングで利用が可能である。
- このボキャブラリーは、エンティティとエンティティ間の関係とアクションをカバーしており、また、よくドキュメント化されている拡張モデルを通じて、容易に拡張できる。
- 一千万以上のサイトが、Webページやe-メールのメッセージをマークアップするのに Schema.orgを利用している。Google, Microsoft, Pinterest, Yandexその他の多くのアプリケーションが、豊かで拡張可能なユーザー経験に力を与えるために、すでに、これらのボキャブラリーを使っている。

エンティティの型(Type)のサンプル

Types = “is-a”

- Book
- BookSeries
- EducationalOrganization
- Event
- GovernmentOrganization
- LocalBusiness
- Movie
- MovieSeries
- MusicAlbum
- MusicGroup
- MusicRecording
- Organization
- Periodical
- Person
- Place
- SportsTeam
- TVEpisode
- TVSeries
- VideoGame
- VideoGameSeries
- WebSite



is-a → Person /people/person

is-a → Film Actor /film/actor

is-a → Politician /government/politician

One entity, many types.

一つのエンティティは、多くの型を持つ

Entityは階層を持つ
schema.org での Entityの階層の例

- Thing > CreativeWork > Book
- Thing > CreativeWork > CreativeWorkSeries > BookSeries
- Thing > Organization > EducationalOrganization
- Thing > Event
- Thing > Organization > GovernmentOrganization
- Thing > Organization > LocalBusiness
  Thing > Place > LocalBusiness
- Thing > CreativeWork > Movie

Entityは階層を持つ
schema.org での Entityの階層の例

- Thing > Place
- Thing > Organization > SportsOrganization > SportsTeam
- Thing > CreativeWork > Episode > TVEpisode
- Thing > CreativeWork > CreativeWorkSeries > TVSeries
  Thing > CreativeWork > TVSeries
- Thing > CreativeWork > Game > VideoGame
  Thing > CreativeWork > SoftwareApplication > VideoGame

schema.org でのPersonのProperty

Properties = “has-a”

- gender
- birthdate
- birthplace
- profession
- nationality
- ethnicity
- parents
- children
- religion
- education
- jobTitle
- **follows**
- **knows**
- **owns**
- **seeks**

... and more

Action

PersonのPropery

- Person: additionalName, address, affiliation, alumniOf, award, awards, birthDate, children, colleague, colleagues, contactPoint, contactPoints, deathDate, email, familyName, faxNumber, follows, gender, givenName, homeLocation, honorificPrefix, honorificSuffix, interactionCount, jobTitle, knows, memberOf, nationality, parent, parents, performerIn, relatedTo, sibling, siblings, spouse, telephone, workLocation, worksFor, ...

The Type Hierarchy
http://schema.org/docs/full.html

PersonのProperty

Properties = “has-a”

Property	Expected Type	Description
Properties from Person		
additionalName	Text	An additional name for a Person, can be used for a middle name.
address	PostalAddress or Text	Physical address of the item.
affiliation	Organization	An organization that this person is affiliated with. For example, a school/university, a club, or a team.
alumniOf	EducationalOrganization or Organization	An organization that the person is an alumni of. Inverse property: alumni.
award	Text	An award won by or for this item. Supersedes awards.
birthDate	Date	Date of birth.
birthPlace	Place	The place where the person was born.
brand	Brand or Organization	The brand(s) associated with a product or service, or the brand(s) maintained by an organization or business person.
children	Person	A child of the person.
colleague	Person or URL	A colleague of the person. Supersedes colleagues.
contactPoint	ContactPoint	A contact point for a person or organization. Supersedes contactPoints.
deathDate	Date	Date of death.
deathPlace	Place	The place where the person died.
duns	Text	The Dun & Bradstreet DUNS number for identifying an organization or business person.

http://schema.org/Person

email	Text	Email address.
familyName	Text	Family name. In the U.S., the last name of an Person. This can be used along with givenName instead of the name property.
faxNumber	Text	The fax number.
follows	Person	The most generic uni-directional social relation.
funder	Organization or Person	A person or organization that supports (sponsors) something through some kind of financial contribution.
gender	GenderType or Text	Gender of the person. While http://schema.org/Male and http://schema.org/Female may be used, text strings are also acceptable for people who do not identify as a binary gender.
givenName	Text	Given name. In the U.S., the first name of a Person. This can be used along with familyName instead of the name property.
globalLocationNumber	Text	The Global Location Number (GLN, sometimes also referred to as International Location Number or ILN) of the respective organization, person, or place. The GLN is a 13-digit number used to identify parties and physical locations.
hasOfferCatalog	OfferCatalog	Indicates an OfferCatalog listing for this Organization, Person, or Service.
hasPOS	Place	Points-of-Sales operated by the organization or person.
height	Distance or QuantitativeValue	The height of the item.
homeLocation	ContactPoint or Place	A contact location for a person's residence.
honorificPrefix	Text	An honorific prefix preceding a Person's name such as Dr/Mrs/Mr.
honorificSuffix	Text	An honorific suffix preceding a Person's name such as M.D. /PhD/MSCSW.
isicV4	Text	The International Standard of Industrial Classification of All Economic Activities (ISIC), Revision 4 code for a particular organization, business person, or place.
jobTitle	Text	The job title of the person (for example, Financial Manager).
knows	Person	The most generic bi-directional social/work relation.

http://schema.org/Person

makesOffer	Offer	A pointer to products or services offered by the organization or person. Inverse property: offeredBy.
memberOf	Organization or ProgramMembership	An Organization (or ProgramMembership) to which this Person or Organization belongs. Inverse property: member.
naics	Text	The North American Industry Classification System (NAICS) code for a particular organization or business person.
nationality	Country	Nationality of the person.
netWorth	MonetaryAmount or PriceSpecification	The total financial value of the person as calculated by subtracting assets from liabilities.
owns	OwnershipInfo or Product	Products owned by the organization or person.
parent	Person	A parent of this person. Supersedes parents.
performerIn	Event	Event that this person is a performer or participant in.
relatedTo	Person	The most generic familial relation.
seeks	Demand	A pointer to products or services sought by the organization or person (demand).
sibling	Person	A sibling of the person. Supersedes siblings.
sponsor	Organization or Person	A person or organization that supports a thing through a pledge, promise, or financial contribution. e.g. a sponsor of a Medical Study or a corporate sponsor of an event.
spouse	Person	The person's spouse.
taxID	Text	The Tax / Fiscal ID of the organization or person, e.g. the TIN in the US or the CIF/NIF in Spain.
telephone	Text	The telephone number.
vatID	Text	The Value-added Tax ID of the organization or person.
weight	QuantitativeValue	The weight of the product or person.
workLocation	ContactPoint or Place	A contact location for a person's place of work.
worksFor	Organization	Organizations that the person works for.

http://schema.org/Person

Intent Model

Alexa$_0$

AlexaのIntent Model

- Alexa は、同じ「意図」を持つ多数の文を、一つのIntentに、結びつける。それは、Cortanaで、一つのコマンド名に、ListenForで記述された複数の入力候補が結びつけられるのと同じである。
- ただ、Alexaでは、Cortanaとはことなり、そのマッピングはプログラム定義とは別のファイル SampleUtterance.txtで行われる。
- このUtteranceデータを通じて、Alexaは。音声をIntentに変えて、ネットをまたいで、デフォールトではクラウド上のユーザーのプログラムに送る。
- ユーザーのプログラムは、このリクエストを受け取り、必要な処理をして、読み上げられる文字列 speech と画面表示用のcardの二つを、レスポンドとして返す。

AlexaのIntent Model

- ユーザー・プログラムは、AWSのLAMBDAに登録することができる。
- プログラムは、イベント・ドリブンのスタイルで書かれる。Android のIntentプログラムの、ネットワーク版だと思っていい。
- プログラムのスタイルは、基本的にVoice Commandと同じ（Speechletと呼ばれた）だが、次第に、ユーザーの質問に対して答えを提供するボイス・サービスとして受け止められるようになる。こうしたサービスをSkillと呼ぶのが、定着し始める。



SampleUtterances.txt

```
HelloIntent こんにちは
HelloIntent はい。こんにちは
HelloIntent おい
HelloIntent こら
HelloIntent なんかいって
HelloIntent お元気ですか
HelloIntent お元気

HelpIntent ヘルプ
HelpIntent 助けて
HelpIntent どうすればいい
HelpIntent お願い
```

入力の発話を待ち受けて、マッチすれば、定義されたIntent Name に対応づける。Intentは、Requestの一部として、Speechletに送られる。

出力用のデータは、プログラム中にハードコードされる。

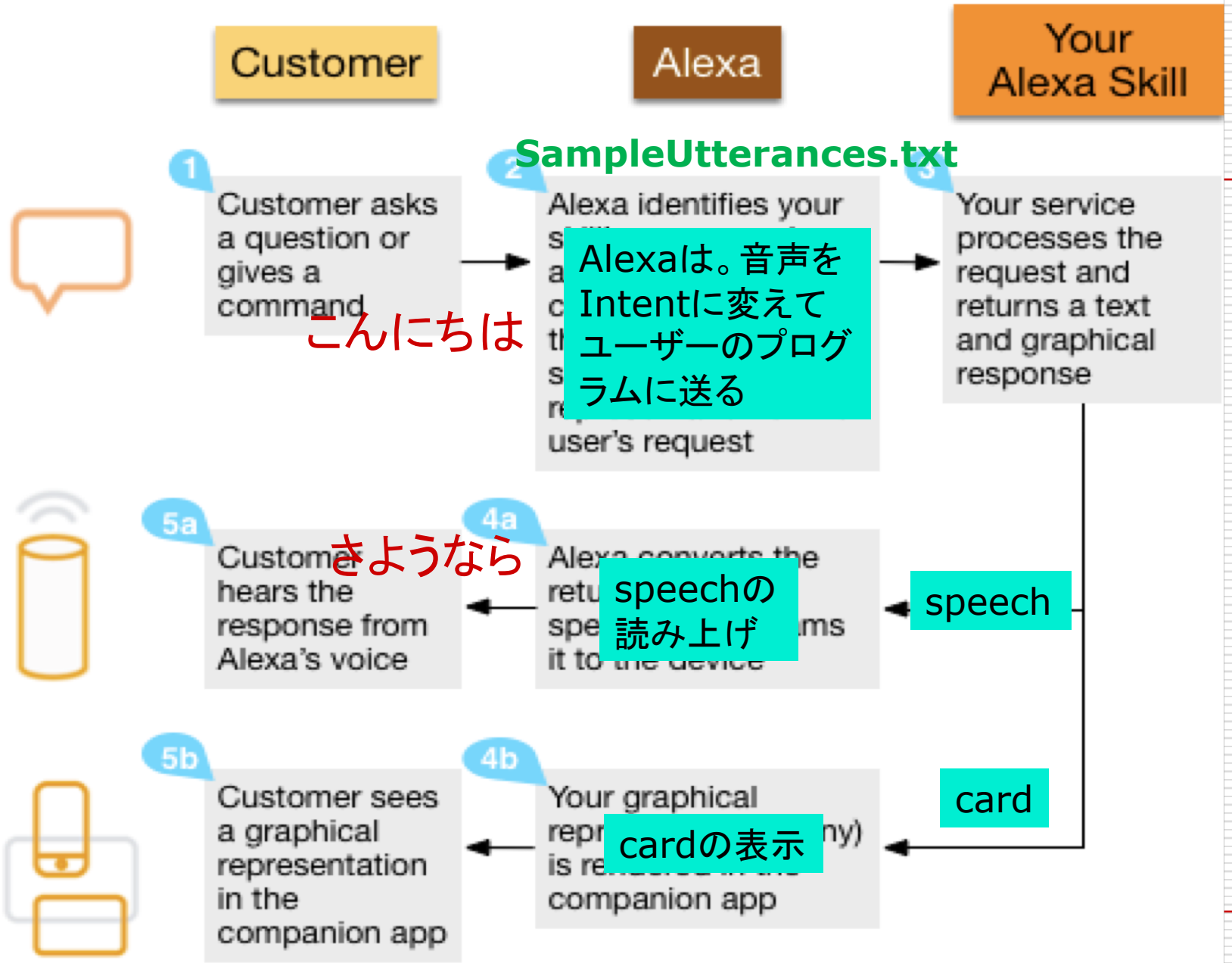



```
public SpeechletResponse onIntent (...) ... {
    ...
    if ("HelloIntent".equals(intentName)) {
        return getHelloResponse();
    } else if ("HelpIntent".equals(intentName)) {
        return getHelpResponse();
    } else {
        throw new SpeechletException("Invalid Intent");
    }
}
```

```
private SpeechletResponse getHelloResponse() {
    String speechText = “さようなら”;
    SimpleCard card = new SimpleCard();
    ...
    return SpeechletResponse.newTellResponse(speech, card);
}
```

音声とカードが返る

ユーザーのプログラム


Customer
Alexa
Your Alexa Skill
SampleUtterances.txt
1
Customer asks a question or gives a command
こんにちは
2
Alexa identifies your
Alexaは。音声をIntentに変えてユーザーのプログラムに送る
user's request
3
Your service processes the request and returns a text and graphical response
speech
speechの読み上げ
4a
5a
Customer hears the response from Alexa's voice
さようなら
card
cardの表示
4b
5b
Your graphical
companion app
Customer sees a graphical representation in the companion app

Intent + Slot Model

Alexa

Alexa History Buff サンプル

User: “Alexa, ask History Buff what happened on August 20th”
Alexa: *(Reads back three events, in reverse chronological order)* “Want to go deeper in history?”
User: “Yes”
Alexa: *(Reads back next set of three events)* “Want to go deeper in history?”
User: “No”

HistoryBuffSpeechlet

```
public class HistoryBuffSpeechlet implements Speechlet {
  ...
  public void onSessionStarted ( ... )  ....
  public SpeechletResponse onLaunch ( ... )  ....
  public SpeechletResponse onIntent ( ... )  ....
  public void onSessionEnded ( ... )  ....

  private SpeechletResponse getWelcomeResponse()  ...
  private Calendar getCalendar( ... ) ...
  private SpeechletResponse handleFirstEventRequest( ... )  ...
  private SpeechletResponse handleNextEventRequest( ... )  ...
  private ArrayList<String> getJsonEventsFromWikipedia( ... ) ...
  private ArrayList<String> parseJson( ... )  ...
  private SpeechletResponse newAskResponse( ... ) ...

}
```

public SpeechletResponse **onIntent**

```
public SpeechletResponse onIntent(final IntentRequest request,
   final Session session) throws SpeechletException {
       log.info("onIntent requestId={}, sessionId={}",
               request.getRequestId(), session.getSessionId());

       Intent intent = request.getIntent();
       String intentName = intent.getName();

       if ("GetFirstEventIntent".equals(intentName)) {
           return handleFirstEventRequest (intent, session);
       } else if ("GetNextEventIntent".equals(intentName)) {
           return handleNextEventRequest (session);
       } else if ("HelpIntent".equals(intentName)) {
          // Create the plain text output.
```

public SpeechletResponse **onIntent**

```
        String speechOutput =
                "With History Buff, you can get"
                        + " historical events for any day of the year."
                        + " For example, you could say today,"
                        + " or August thirtieth, or you can say exit. Now, “
                        + “which day do you want?”;
        String repromptText = "Which day do you want?”;
        return newAskResponse(speechOutput, repromptText);

    } else if ("FinishIntent".equals(intentName)) {
        PlainTextOutputSpeech outputSpeech =
            new PlainTextOutputSpeech();
        outputSpeech.setText("Goodbye");
        return SpeechletResponse.newTellResponse(outputSpeech);
    } else {
        throw new SpeechletException("Invalid Intent");
    }
}
```

historybuff.speechAsset.
IntentSchema.jason

```
{
  "intents": [
    {
      "intent": "GetFirstEventIntent",
      "slots": [
        {
          "name": "day",
          "type": "DATE"
        }
      ]
    },
    {
      "intent": "GetNextEventIntent",
      "slots": []
    },
```

historybuff.speechAsset.
IntentSchema.jason

```
    {
      "intent": "HelpIntent",
      "slots": []
    },
    {
      "intent": "FinishIntent",
      "slots": []
    }
  ]
}
```

historyBuff.speechAsset.
SampleUtterances.txt

GetFirstEventIntent get events for {august seventeenth|day}
GetFirstEventIntent get events for {december thirty first|day}
GetFirstEventIntent get events for {next monday|day}
GetFirstEventIntent get events for {today|day}
GetFirstEventIntent give me events for {august seventeenth|day}
GetFirstEventIntent give me events for {next monday|day}
GetFirstEventIntent give me events for {today|day}
GetFirstEventIntent what happened on {august seventeenth|day}
GetFirstEventIntent what happened on {next monday|day}
GetFirstEventIntent what happened on {today|day}
GetFirstEventIntent what happened
GetFirstEventIntent {january first|day}
GetFirstEventIntent {january second|day}
GetFirstEventIntent {january third|day}
GetFirstEventIntent {january fourth|day}
GetFirstEventIntent {january fifth|day}
GetFirstEventIntent {january sixth|day}
GetFirstEventIntent {january seventh|day}

historyBuff.speechAsset.
SampleUtterances.txt

GetFirstEventIntent {january eighth|day}
GetFirstEventIntent {january ninth|day}
GetFirstEventIntent {january tenth|day}
GetFirstEventIntent {january eleventh|day}
GetFirstEventIntent {january twelfth|day}
GetFirstEventIntent {january thirteenth|day}
GetFirstEventIntent {january fourteenth|day}
GetFirstEventIntent {january fifteenth|day}
GetFirstEventIntent {january sixteenth|day}
GetFirstEventIntent {january seventeenth|day}
GetFirstEventIntent {january eighteenth|day}
GetFirstEventIntent {january nineteenth|day}
GetFirstEventIntent {january twentieth|day}
GetFirstEventIntent {january twenty first|day}
GetFirstEventIntent {january twenty second|day}
GetFirstEventIntent {january twenty third|day}
GetFirstEventIntent {january twenty fourth|day}
GetFirstEventIntent {january twenty fifth|day}

historyBuff.speechAsset.
SampleUtterances.txt

GetFirstEventIntent {january twenty sixth|day}
GetFirstEventIntent {january twenty seventh|day}
GetFirstEventIntent {january twenty eighth|day}
GetFirstEventIntent {january twenty ninth|day}
GetFirstEventIntent {january thirtieth|day}
GetFirstEventIntent {january thirty first|day}
GetFirstEventIntent {february first|day}
GetFirstEventIntent {february second|day}
GetFirstEventIntent {february third|day}
GetFirstEventIntent {february fourth|day}
GetFirstEventIntent {february fifth|day}
GetFirstEventIntent {february sixth|day}
GetFirstEventIntent {february seventh|day}
GetFirstEventIntent {february eighth|day}
GetFirstEventIntent {february ninth|day}
GetFirstEventIntent {february tenth|day}
GetFirstEventIntent {february eleventh|day}
GetFirstEventIntent {february twelfth|day}

historyBuff.speechAsset.
SampleUtterances.txt

```
GetFirstEventIntent {february thirteenth|day}
GetFirstEventIntent {february fourteenth|day}
GetFirstEventIntent {february fifteenth|day}
GetFirstEventIntent {february sixteenth|day}
GetFirstEventIntent {february seventeenth|day}
 ....
 ....
GetFirstEventIntent {december twenty third|day}
GetFirstEventIntent {december twenty fourth|day}
GetFirstEventIntent {december twenty fifth|day}
GetFirstEventIntent {december twenty sixth|day}
GetFirstEventIntent {december twenty seventh|day}
GetFirstEventIntent {december twenty eighth|day}
GetFirstEventIntent {december twenty ninth|day}
GetFirstEventIntent {december thirtieth|day}
GetFirstEventIntent {december thirty first|day}
```

historyBuff.speechAsset.
SampleUtterances.txt

```
GetFirstEventIntent {next Monday|day}
GetFirstEventIntent {next Tuesday|day}
GetFirstEventIntent {next Wednesday|day}
GetFirstEventIntent {next Thursday|day}
GetFirstEventIntent {next Friday|day}
GetFirstEventIntent {next Saturday|day}
GetFirstEventIntent {next Sunday|day}
GetFirstEventIntent {today|day}
GetFirstEventIntent {tomorrow|day}
GetFirstEventIntent {yesterday|day}

FinishIntent no
FinishIntent nope
FinishIntent no thanks
FinishIntent no thank you
```

historyBuff.speechAsset.
SampleUtterances.txt

HelpIntent helpHelpIntent help me
HelpIntent what can I ask you
HelpIntent get helpHelpIntent to help
HelpIntent to help me
HelpIntent what commands can I ask
HelpIntent what commands can I say
HelpIntent what can I do
HelpIntent what can I use this for
HelpIntent what questions can I ask
HelpIntent what can you do
HelpIntent what do you do
HelpIntent how do I use you
HelpIntent how can I use you
HelpIntent what can you tell me
GetNextEventIntent yes
GetNextEventIntent yup
GetNextEventIntent sure
GetNextEventIntent yes please

Intent と Slot
Slot Typeの導入

Intent と Slot

- Intentは、ユーザによって話されたリクエストを実現するActionを表す。
- Intentは、オプションで引数を持つことができる。その引数をSlotという。SampleUtteranceに、Slotの記述を許す。

```
IntentName this is a sample utterance with no slots
IntentName this is a sample utterance containing a {SlotName}
IntentName this is a sample utterance containing a {SlotName} and {AnotherSlotName}
```

- 旅行を計画するスキルのIntentの名前を **PlanMyTrip** とすれば、PlanMyTripは、**toCity, fromCity, travelDate** といったSlotを持つだろう。

Intent PlanMyTripのスキーマ定義

```
{
  “intents” : [
     {
        “intent” : “PlanMyTrip” ,
        “slots”   : [
           {
              “name”: “fromCity” ,
              ”type”  : “AMAZON.US_CITY”
           } ,
           {
              “name”: “toCity” ,
              ”type”  : “AMAZON.US_CITY”
           } ,
           {
              “name”: “travelDate” ,
              ”type”  : “AMAZON.DATE”
           }
        ]
     }
}
```

Intent GetHoroscopeのスキーマ定義の例

```
{
  "intents": [
    {
      "intent": "GetHoroscope",
      "slots": [
        {
          "name": "Sign",
          "type": "LIST_OF_SIGNS"
        },
        {
          "name": "Date",
          "type": "AMAZON.DATE"
        }
      ]
    }
  ]
}
```

Custom slot type
LIST_OF_SIGNS の定義例

```
Aries
Taurus
Gemini
Cancer
Leo
Pisces
Virgo
Libra
Scorpio
Sagittarius
Capricorn
Aquarius
```

対応する sample utterances:

```
GetHoroscope what is the horoscope for {Sign}
GetHoroscope what will the horoscope for {Sign} be {Date}
GetHoroscope what will the horoscope for {Sign} be on {Date}
GetHoroscope get me my horoscope
GetHoroscope {Sign}
...
(many more sample utterances)
```

User: Alexa, ask *Plan My Trip* to plan a trip from Seattle to Portland on Friday.

上のような発話がなされたとする。
Utteranceに次のような記述があったとしよう。

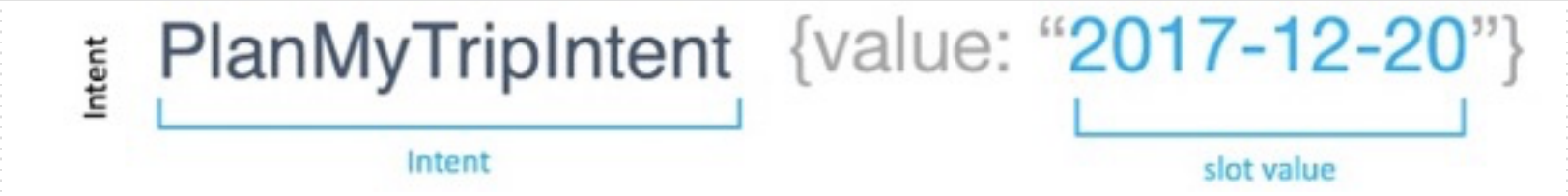
Plan a trip from {fromCity} to {toCity} on {travalDate}

Alexa は、fromCity スロットに ”Seatle“、とCity スロットに ”Portland“、travelDate スロットに”Friday“ という値を、PlayMyTrip Intentの引数として渡して、”Plan My Trip”サービスを呼び出す。

発話



Skillの呼び出し



SlotとEntity Resolution

人は、よく、同じ物事を違う言葉で話すことがある。
MEDIA_TYPEというslotは、mediaの特定の型、例えば、”song”, “track”, “single” と言ったものを集めるようとする。
たいていの場合、コード的には、この三つの表現を同じように扱うことができる。

MEDIA_TYPEを、一つのEntity として、その取りうる値を、もっと正確に定義することができる。それを、Entity Resolutionという。

Entity Resolution では、Slot Typeの値には、IDがつけられ、かつ、その同義語（synonym）のリストが追加される。話者が話した発話自体は、”name” の ”value” とされる。

Slotの型のEntity Resolution

```
{
  "types": [
    {
      "name": "MEDIA_TYPE",
      "values": [
        {
          "id": "ALBUM",
          "name": {
            "value": "album",
            "synonyms": ["cd","lp","record"]
          }
        },
        {
          "id": "SONG",
          "name": {
            "value": "song",
            "synonyms": ["track","single","tune"]
          }
        }
      ]
    }
  ]
}
```

Slot Type	Short Description	Supported Languages
AMAZON.DATE	Converts words that indicate *dates* ("today", "tomorrow", or "july") into a date format (such as "2015-07-00T9").	English (UK) English (US) German
AMAZON.DURATION	Converts words that indicate *durations* ("five minutes") into a numeric duration ("PT5M").	English (UK) English (US) German
AMAZON.FOUR_DIGIT_NUMBER	Converts words that represent *four-digit numbers* ("six oh four five" into digits ("6045").	English (UK) English (US) German
AMAZON.NUMBER	Converts numeric words ("five") into digits (such as "5").	English (UK) English (US) German
AMAZON.TIME	Converts words that indicate time ("four in the morning", "two p m") into a time value ("04:00", "14:00").	English (UK) English (US) German

Slot Type

Numbers, Dates, and Times

AMAZON.Actor
AMAZON.AdministrativeArea
AMAZON.AggregateRating
AMAZON.Airline
AMAZON.Airport
AMAZON.Animal
AMAZON.Artist
AMAZON.AT_CITY
AMAZON.AT_REGION
AMAZON.Athlete
AMAZON.Author
AMAZON.Book
AMAZON.BookSeries
AMAZON.BroadcastChannel
AMAZON.CivicStructure
AMAZON.Color
AMAZON.Comic
AMAZON.Corporation
AMAZON.Country
AMAZON.CreativeWorkType
AMAZON.DayOfWeek
AMAZON.DE_CITY
AMAZON.DE_FIRST_NAME
AMAZON.DE_REGION
AMAZON.Dessert
AMAZON.DeviceType
AMAZON.Director
AMAZON.Drink
AMAZON.EducationalOrganization
AMAZON.EUROPE_CITY
AMAZON.EventType
AMAZON.Festival
AMAZON.FictionalCharacter
AMAZON.FinancialService
AMAZON.Food
AMAZON.FoodEstablishment
AMAZON.Game
AMAZON.GB_CITY
AMAZON.GB_FIRST_NAME
AMAZON.GB_REGION
AMAZON.Genre
.....
.....
AMAZON.WrittenCreativeWorkType

Slot Type

List Type

Intent Signature

Schema.orgのEntity Modelの採用

```
Action<propertyName@EntityType[propertyName]>
```

Built-in Intent Library

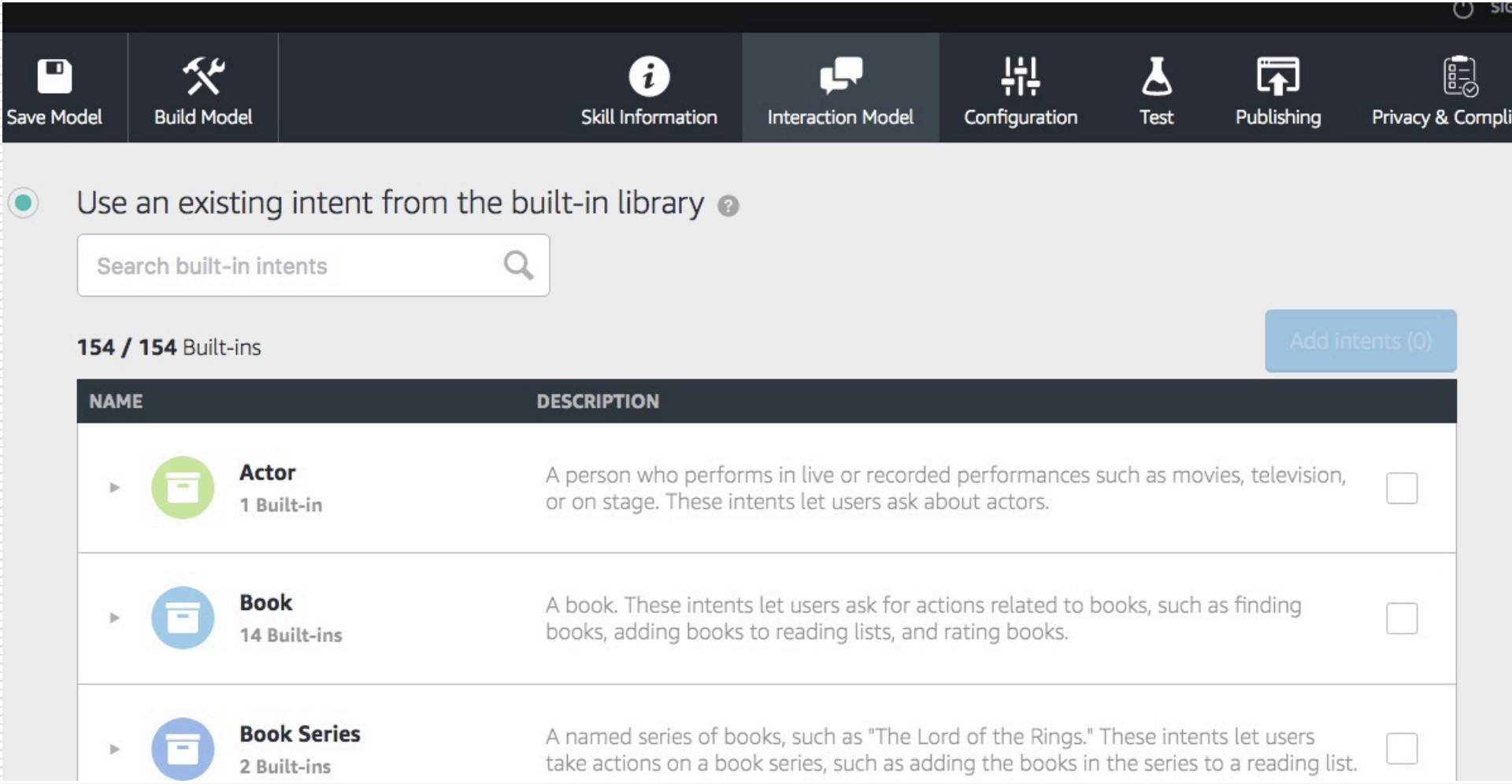
Alexa Skills Kit は、あらかじめ用意されたビルドインのIntent のライブラリーを提供する。ユーザは、発話のサンプルを用意することなく、このIntentの機能をSkillの中で利用することができる。

https://goo.gl/Dw39Bg

built-in intent libraryとschema.org

The built-in intent library incorporates material from Schema.org, which is licensed under the Creative Commons Attribution-ShareAlike License (version 3.0) (the "License"). You may not use this file except in compliance with the License. You may obtain a copy of the License at http://creativecommons.org/licenses/by-sa/3.0/. For questions, please reach out to alexa-ontology-support@amazon.com.

次のような、build-in Intentが用意されている



Actor 1 Built-in	A person who performs in live or recorded performances such as movies, television, or on stage. These intents let users ask about actors.	☐
Book 14 Built-ins	A book. These intents let users ask for actions related to books, such as finding books, adding books to reading lists, and rating books.	☐
Book Series 2 Built-ins	A named series of books, such as "The Lord of the Rings." These intents let users take actions on a book series, such as adding the books in the series to a reading list.	☐
Calendar 2 Built-ins	A calendar that contains a list of events. These intents let users make requests related to calendars, such as reading back events, looking up events, and adding items to a calendar.	☐
Catalog 1 Built-in	A collection or library offered and curated by an organization or collected and organized by an individual. These intents let users ask about collections of books or music.	☐
Creative Work 3 Built-ins	A generic kind of creative work, including books, movies, photographs, software programs, and so on. These intents let users make requests related to any type of creative work.	☐
Creative Work Section 8 Built-ins	A section or part of a creative work, such as a page or chapter. These intents let users specify a particular part of a book, movie, page, or other section.	☐

Creative Work Section 8 Built-ins	A section or part of a creative work, such as a page or chapter. These intents let users specify a particular part of a book, movie, page, or other section.	☐
Episode 6 Built-ins	An episode for TV, radio, video game, or other media which can be part of a series or season. These intents let users request actions related to episodes, such as starting playback of a particular episode or searching for a particular episode.	☐
Event 8 Built-ins	An event happening at a certain time and location, such as a meeting, concert, lecture, or festival. These intents let users make requests related to events, such as adding an event to a calendar, or looking up details of an event.	☐
Local Business 4 Built-ins	A particular physical business or branch of an organization, like a restaurant, a medical practice, or a store. These intents let users ask for information about businesses such as the location, contact information, or opening hours.	☐
Musician 1 Built-in	A person or group that performs music. These intents let users request actions related to musicians, such as asking to rate or "like" a musician.	☐
Music Creative Work 14 Built-ins	The broad category of creative works that are musical in nature, includes songs, albums, playlists, and other items. These intents let users request actions related to musical works, such as adding songs to a library, controlling playback of musical works, and looking up information about musical works.	☐
Music Group 2 Built-ins	A musical group, such as a band, an orchestra, or a choir. This can also represent a solo musician. These intents let users request actions related to a music group, such as "liking" a group, finding songs by a group, or asking about the members of a	☐

Music Playlist 10 Built-ins	A collection of music tracks in playlist form. These intents let users make requests to manipulate playlists by adding, removing, and choosing items.	☐
Music Recording 5 Built-ins	A music recording (track), usually a single song. These intents let users ask for specific information about songs, such as the artist who performs the song, the genre of the song, or other information.	☐
Playback Mode 2 Built-ins	Represents modes related to playback, such as shuffle and loop. These intents let users ask to turn these modes on and off.	☐
Reading List 7 Built-ins	A list made up of books or other written creative works that a user wants to collect in one place, such as a summer reading list or a "read later" list. These intents let users make requests about reading lists, such as creating lists and adding or removing items on the lists.	☐
Screening Event 4 Built-ins	A screening of a movie or other video. These intents let users make requests related to particular screening events, such as asking for the show time for an event, asking what movies are playing, and asking what movie theaters are playing a particular movie.	☐
Singer 1 Built-in	A person who sings, either as a solo performer or as part of a group. These intents let users request actions related to singers, such as asking who sings a particular song.	☐
Standard 22 Built-ins	The standard intents are used for common, general actions such as stopping, canceling, and asking for help.	☐

Thing 5 Built-ins	A generic object. These intents let users make requests that are not related to any specific object or entity, such as requests to pause or continue an action.	☐
TV Season 1 Built-in	Season during which a series of related episodes are broadcast on TV broadcast or delivered online. These intents let users request actions for a TV season, such asking to start playback of a particular season for a show.	☐
TV Series 2 Built-ins	A series of related television episodes. These intents let users request actions for a TV series, such asking to start playback of a particular series.	☐
Video Creative Work 23 Built-ins	A creative work that includes video and possibly audio, such as movies, television shows, and video clips. These intents let users request actions related to video works, such as adding videos to a library or watchlist, controlling playback of videos, and looking up information about video works.	☐
Video Playlist 3 Built-ins	A collection of video creative works, such as a list of shows to watch. These intents let users make requests to manipulate video playlists by adding, removing, and choosing items.	☐
Weather Forecast 3 Built-ins	A weather forecast. These intents let users ask for general weather forecasts, as well as more specific questions about specific weather conditions.	☐

Understanding the Structure of the Built-in Intent Library

https://developer.amazon.com/public/solutions/alexa/alexa-skills-kit/docs/understanding-the-structure-of-the-built-in-intent-library

Intent Signature Syntax

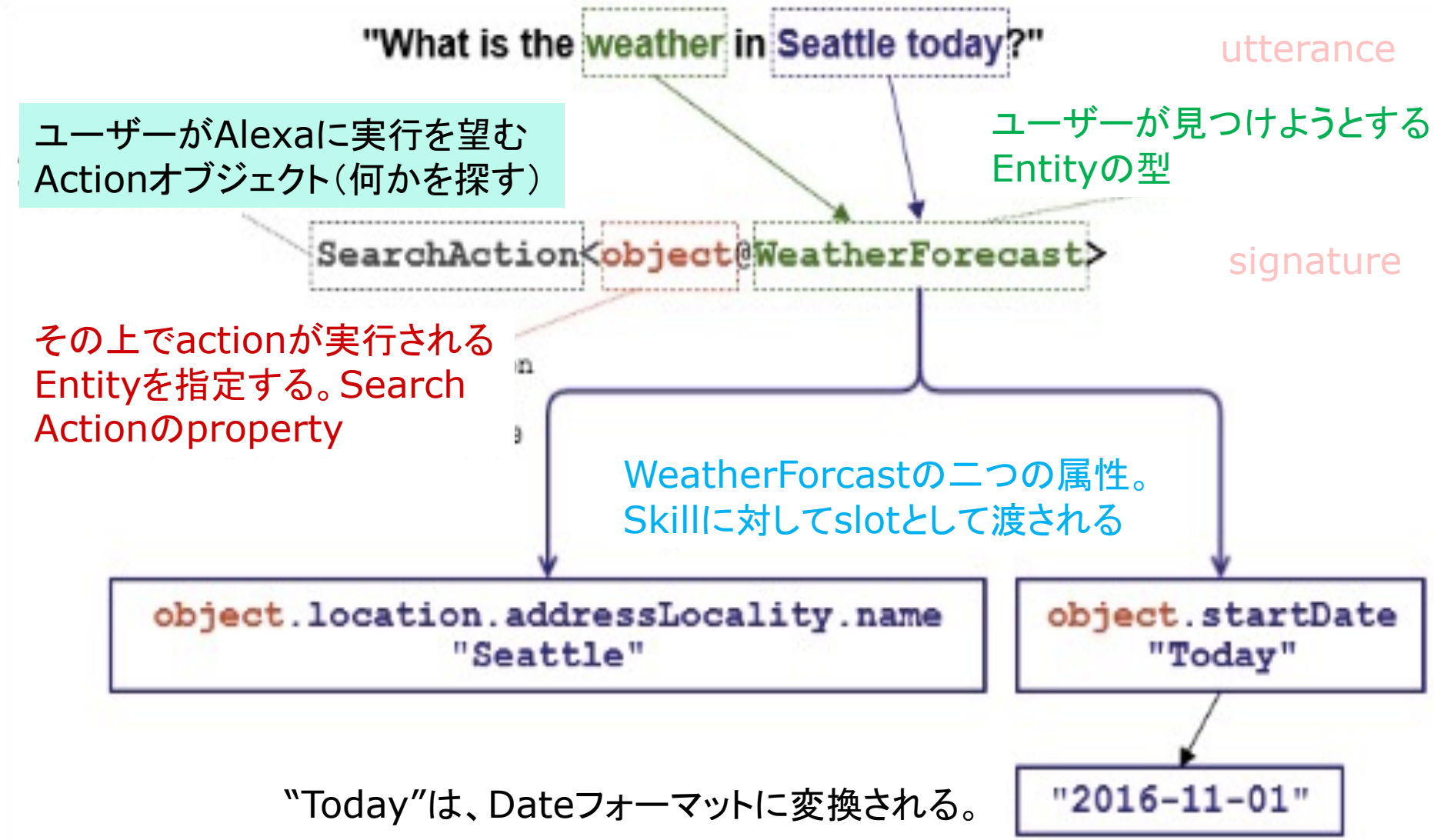
- built-in intent は、***intent signature***を持つ。このsignatureを、intentの名前としてSchemaに書くことができる。次のサンプルは、二つのWeatherForecast intents をもった、Schemaの例である。

```
{
  "intents": [
    {
      "intent": "AMAZON.SearchAction<object@WeatherForecast>"
    },
    {
      "intent": "AMAZON.SearchAction<object@WeatherForecast[temperatu
    }
  ]
}
```

- **AMAZON**: all built-in intents begin with the AMAZON namespace.
- **SearchAction**: an ***action*** that identifies the action the user wants Alexa to take, such as searching for information.
- **object**: a **property of SearchAction**, for identifying the **entity** on which it should operate.
- **WeatherForecast**: the **entity type** for object. This identifies the entity the user wants to search for.
- **[temperature]:** (optional) if included, specifies an attribute to narrow the scope of the intent. This creates an intent that is triggered when the utterance mentions the specified attribute.

For instance, the intent AMAZON.SearchAction<object@WeatherForecast[temperature]>would be triggered by "what's today's temperature," but not the more generic "what's the weather today."

Utterance と Signature



```
{
  "intents": [
    {
      "intent": "AMAZON.SearchAction<object@WeatherForecast>"
    }
  ]
}
```

- このSchemaには、slotが定義されていないのだが、“what’s the weather **today** in **Seattle**” のような発話は、今日の日付と都市 “Seatle”をSlotに含んだRequestをSkillに送る。
- これらのIntentは、**Actionと、Entityの型と、プロパティ**から構成されている。個々のIntentの名前は（WeatherForcastは、Built-in-Intentの名前である ）、これらの要素と一緒に結合されて一つの**Intent Signature**（例えば AMAZON.SearchAction<object@Weather Forecast>といった）にまとめられている。このSignatureをIntentの名前として利用できる。

Actions, Entity Types, and Properties

- An ***action*** represents an action the user is trying to take, such as searching for information.
- For instance, **SearchAction** represents the "searching for information" action. An intent defined with SearchAction indicates that the user wants search or look up information, such as finding a weather forecast or finding the phone number for a nearby coffee shop.
- An intent defined with **AddAction** indicates that the user wants to add information somewhere, such as adding an event to a calendar or adding the title a book to a reading list.

Actions, **Entity Types**, and **Properties**

- Most **actions** operate *on* other objects or *use* other objects. A search action needs to identify the information to search for. An add action needs to identify the item to add and possibly also the list to which to add it.
- These items are specified using the ***properties*** of the action.
- For example, SearchAction has an **object** property that identifies the ***type*** of information the user wants to find.
- AddAction has a **targetCollection** property to identify the **type** of the target list and an object property to identify the **type** object to add to the list.

Actions, **Entity Types**, and Properties

- An ***entity type*** represents the objects that the actions can act on or use.
- **Entity types** also have **attributes**, which are represented by **properties**.
- For example, WeatherForecast is an **entity type** that defines a weather forecast. It has properties such as location, duration, and temperature (among others).

```
Action<propertyName@EntityType[propertyName]>
```

Identifying the parts of an utterance

- The entire utterance indicates that the user wants to *search for something* or *find information*. **Action**
- The "object" of the search – the information the user wants to find – is a *weather forecast*. This is the main *focus* or *purpose* in the sentence. **Entity**
- The words "Seattle" and "today" are attributes of the weather forecast the user wants to find. **Property**



SearchActionクラスは
情報を探すという意図を
表現する

SearchActionのobject
プロパティは、見つけるべき
Entityを表現する



WeatherForcastクラスは
天気予報を表現する

それは、場所（"Seatle"）や
日付（"Today"）といった属性を
表現しているプロパティを持つ

AMAZON.SearchAction<object@WeatherForecast>

このコンテキストでは、@サインは、object のプロパティは、
WeatherForecast型を持つことを宣言している

"Search for Weather Forecast."

複数のプロパティを持つAction

AMAZON.AddAction<object@Book,
targetCollection@ReadingList>

- このコンテキストでも、同様に、@サインは、**object**プロパティは**Book**型を持ち、**targetCollection**プロパティは**ReadingList**型を持つことを宣言している。

- この intent signatureは、次のように読める。

Add Book to Reading List.

Example: Creating a Weather Skill with the WeatherForecast Intents

- A general request for the weather:

  User: Alexa, ask My Weather what's the weather in Seattle today?
  My Weather: Light rain this afternoon through tomorrow, with temperatures falling to 56 degrees on Monday.

- A request for the expected temperature:

  User: Alexa, ask My Weather what will the high temperature be in Seattle today?
  My Weather: Today, expect a high of 65 degrees, and a low of 49.

- A question about a particular weather condition:

  User: Alexa, ask My Weather when will it rain next in Seattle?
  My Weather: Light rain is expected to start this afternoon, continuing until evening.

AIを巡る新たな社会的問題



Cambridge Analytica社による選挙干渉疑惑



https://goo.gl/WVwDw1



"The reality here is that if you want to build a service that helps connect everyone in the world, then **there are a lot of people can't afford to pay**," Zuckerberg said. "And therefore, as with a lot of media, having an **advertising-supported model is the only rational model that can support building this service to reach people**."

https://goo.gl/EJ9s8V

AIを巡る新たな社会的問題



2000年

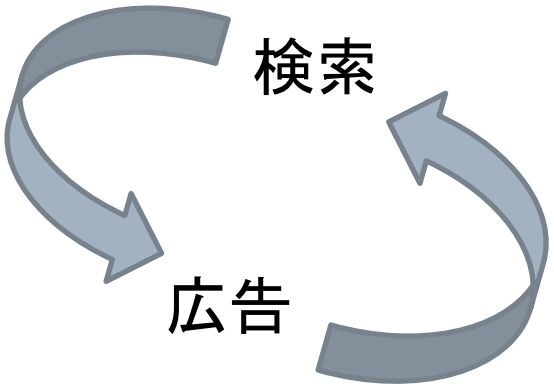
リスティング広告
Page Rank
Hadoop

2010年

検索のリアルタイム化
Knowledge Graph
音声検索

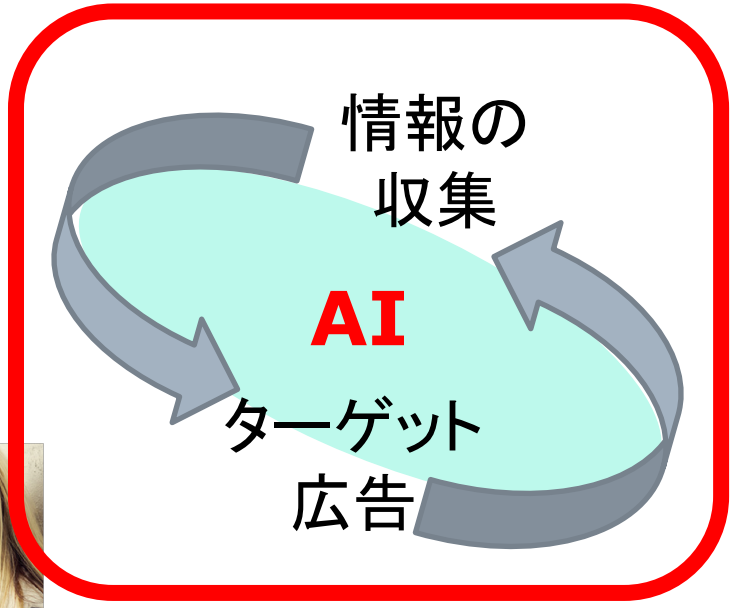
2020年

情報の収集・ターゲット広告へのAIの利用









Googleの
大規模分散システム

SNS、
スマートフォン、
クラウドの登場

AI技術の利用の拡大

Part III

計算主義的言語理論
Minimalist ProgramとCategorial Grammar



Part III
計算主義的言語理論　概説
Minimalist ProgramとCategorial Grammar

- 言語に対する現代の計算主義的なアプローチは、人工知能研究と同じくらい古い。Church-Turingによる「計算可能性」の定式化やTuringによる「機械は考えることができるか？」という問題提起が行われた時代の1950年代に、Chomskyは登場する。Chomsky Hierarchyの論文が出たのは1956年、”Syntactic Structures”が出るのは1957年だ。
- 1995年、Chomskyの”Minimalist Program”が出版される。それは、それまでの彼自身の言語理論をも一新したものだった。それは、それ以降の文法理論に深い影響を与えることになる。”Minimalist”の名前は、必要最小限の構成要素から文法理論を再構成しようというアプローチを示すものだが、その直接のソースは、”Principles of minimal computation”（「最少計算の原理」）から来ている。彼は、最も簡単な計算操作は、二つのものを一つに合わせる”Merge” だとする。

Part III

計算主義的言語理論　概説

Minimalist ProgramとCategorial Grammar

- 言語能力を計算能力として捉えようとするときに、その中心的な課題は、文法的に正しい文を生成する計算規則を見つけることだ。それは、文法の計算ルールを見つけることだと言って良い。MinimalistのChomskyは、それをMergeだとする。
- 時計の針を、少し、巻き戻そう。
  60年ほど前の1958年に、Chomskyの僚友だったLambekは、驚くべき発見をする。文法の計算ルールは、次のたった二つの式で表されるというのだ。

  (x/y)y --> x
  y(y\x) --> x

- Lambekは、名詞を表す型nと、文を表す型sというたった二つの型を用いて、語の並びから、先の二つの計算ルールで文を導く計算をしてみせる。Categorial Grammarの誕生である。

Part III

計算主義的言語理論　概説
Minimalist ProgramとCategorial Grammar

- 1958年の論文の後、Lambekは数学の世界で研究を続けることになる。ところが、それから50年経った2008年、Lambekは興味ふかい言語学の論文 “From Word to Sentence” を発表する。
- なぜ彼は50年経って、また言語学に興味を覚えたのか？
  その理由は、明らかだと思う。かつての僚友 Chomskyが、1995年に発表し、現在の言語学の大きな潮流となった "Minimalist Program"とその"Merge"という基本コンセプトに大きな刺激を受けたからだと僕は考えている。
- 長い時間がかかったのだが、現在では、計算主義的言語理論の二つの大きな潮流である、Minimalist ProgramとCategorial Grammarとの融合が進んでいるように見える。

Part III　Agenda

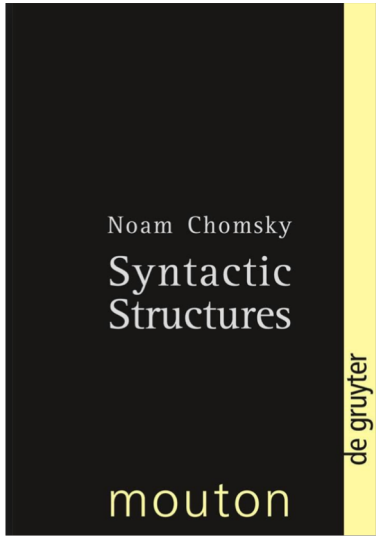
計算主義的言語理論
Minimalist ProgramとCategorial Grammar

- Minimalist ProgramとCategorial Grammar
- 文法の階層性
  -- Chomsky Hierarchyについて
- Minimalist Programの言語像
  -- 普遍文法をボトムアップで構成する
- Mergeとは何か？
- 計算システムを構成するもの
- 計算システムとしての言語能力
  Language = Interfaces + Recursion
- Categorial Grammar -- Lambek（1958）
- Word to Sentence -- Lambek（2008）

Minimalist Programと Categorial Grammar



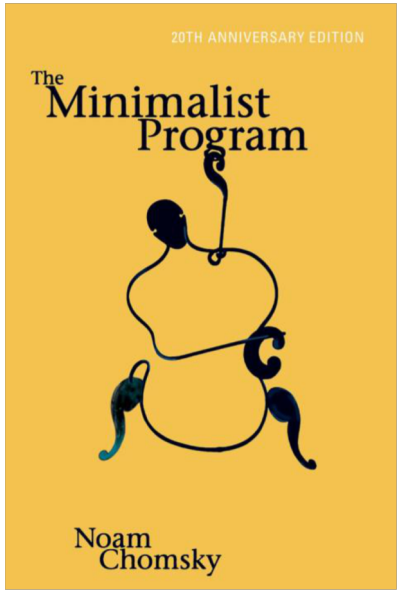
1957年



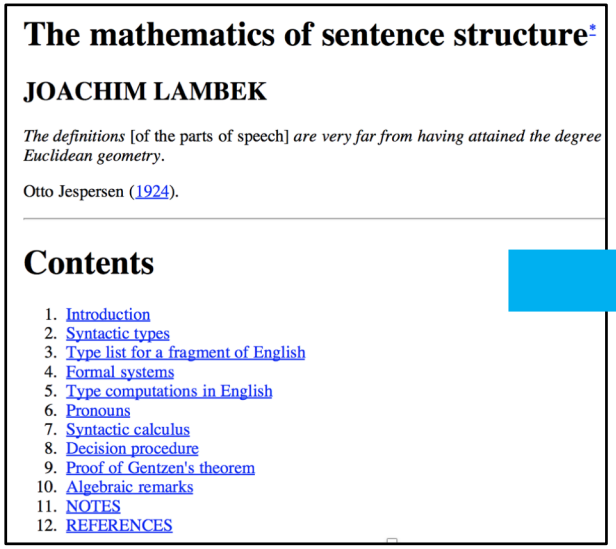
Chomsky

48年

1995年



1958年



The mathematics of sentence structure*

JOACHIM LAMBEK

The definitions [of the parts of speech] *are very far from having attained the degree Euclidean geometry.*

Otto Jespersen (1924).

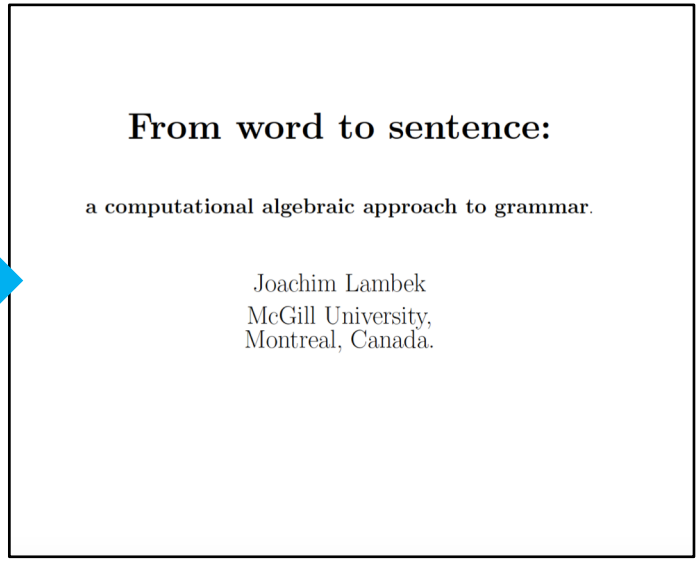
Contents

1. Introduction
2. Syntactic types
3. Type list for a fragment of English
4. Formal systems
5. Type computations in English
6. Pronouns
7. Syntactic calculus
8. Decision procedure
9. Proof of Gentzen's theorem
10. Algebraic remarks
11. NOTES
12. REFERENCES

Lambek

50年！

2008年



From word to sentence:

a computational algebraic approach to grammar.

Joachim Lambek
McGill University,
Montreal, Canada.

Minimalist文法とカテゴリー文法の収斂について

1 At the Cartesian Well: The Minimalist Program & Categorial Grammar

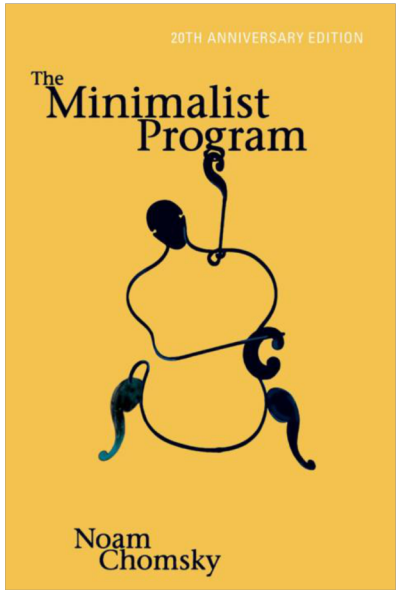
Imagine the following scene. You are at your favorite beer hall somewhere in Amsterdam—let's call it the Cartesian Well. Well known meeting place of intelligentsia, you are not surprised when a thin person dressed all in black sidles up to you and whispers in your ear, "Have I got a linguistic theory for you!" You of course yawn, have heard many such fables in your time; besides you have drunk too much. "No, wait," the figure grabs your shoulder, "I've discovered that Chomsky's latest approach to syntax and categorial grammar are *converging*."

Another flat-earth cultist? you think. "Well, hear me out—at least let me buy you another beer." So you to listen to the tale:

4 Conclusions from the Beer Hall

To summarize, given the current push towards "minimalism" in the so-called government-and-binding approach seems to have eliminated both government (and binding, not discussed here) in favor of a single hierarchical concatenation operator that meshes perfectly with the classical theory of categorial grammar, as well as providing a natural *explanation* for the observed grammatical relations and a transparent framework on which to build a model of sentence processing. While this trend surely does not solve all our "religious" problems, it certainly goes a long way towards taking down the "barriers" to a mature, ecumenical framework within which to reach common ground among what has long appeared to be quite disparate accounts of natural language. Perhaps we can all now drink beer together.

1995年 Berwick & Epstein
"On the Convergence of 'Minimalist' syntax and Categirial Grammar"



1995年

On the Convergence Of 'Minimalist' Syntax and Categorial Grammar

Professor Robert C. Berwick, MIT Department of Computer Science

Professor Samuel David Epstein, Harvard Department of Linguistics
Cambridge, MA 02139

ABSTRACT

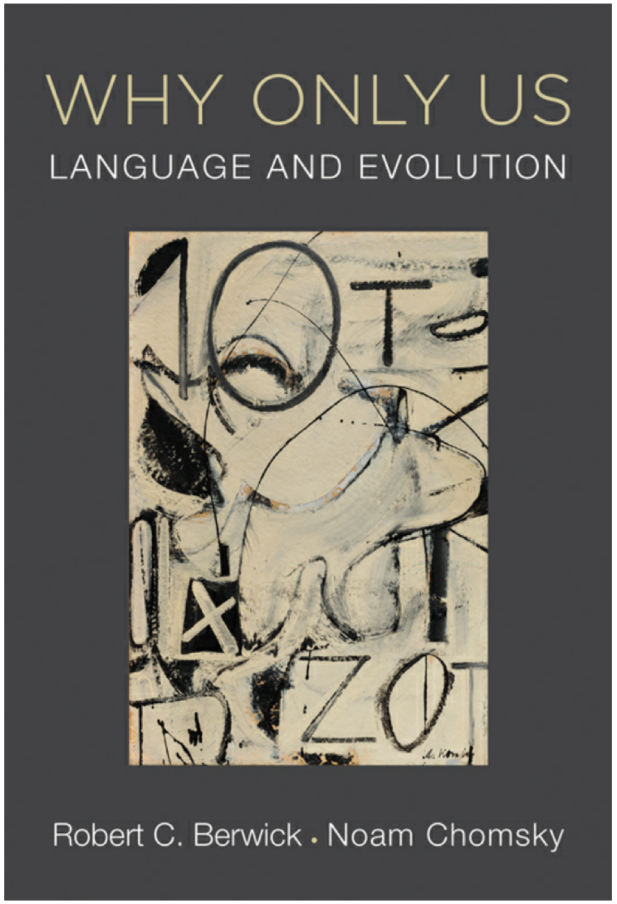
This paper shows that the so-called "Minimalist Program" of Chomsky (1993, 1995) can be given a natural interpretation as a categorial system in which there is exactly one syntactic (algebraic) operation: namely, "Hierarchically Concatenate" (HC) (what Chomsky calls "Merge"), and also replacing the representations of "D-structure," 'S-structure" and transformations with the derivation lines typical of categorial systems—thus unifying two previously disparate approaches to the analysis of natural language. For example, the general "movement" rule of transformational grammar is easily seen to be a subcase of Hierarchical Concatenation of (alpha, beta), where alpha is a subtree of beta; this automatically derives the usual c-command condition on so-called "empty categories." The usual semantic interpretation benefits of the categorial

terpret" sentences.

1 AT THE CARTESIAN WELL: THE MINIMALIST PROGRAM & CATEGORIAL GRAMMAR

Imagine the following scene. You are at your favorite beer hall somewhere in Amsterdam—let's call it the Cartesian Well. Well known meeting place of intelligentsia, you are not surprised when a thin person dressed all in black sidles up to you and whispers in your ear, "Have I got a linguistic theory for you!" You of course yawn, have heard many such fables in your time; besides you have drunk too much. "No, wait," the figure grabs your shoulder, "I've discovered that Chomsky's latest approach to syntax and categorial grammar are *converging*."

2016年



Berwick & Chomsky

Lambek: "Remarks on the history of categorial grammar"

- A few years later, I was invited to a symposium of the American Mathematical Society [1961] on applications of mathematics to linguistics. Hard pressed to find anything new to say, I discussed a non-associative version of the syntactic calculus.
- These ideas did not catch on with the linguistic community and even met with some antagonism by certain graduate students, although two linguists, Noam Chomsky and Robert Lees, tried to encourage me to continue. Still, I could see the new wave of Chomsky's generative-transformational grammar sweeping everything else aside and turned my attention to other things.

https://goo.gl/T7vHsh

Lambek:"Word to Sentence"
0. Preface.

- Chomsky refined and deepened his theory in subsequent years, publishing almost one book per annum. Yet, while going profoundly into the structure of natural languages, it became less accessible to computational implementation, at least in the opinion of some categorial linguists. Prompted by a suggestion by Claudia Casadio, I proposed a new computational algebraic approach in 1998, founded in the categorial tradition and based on a slight generalization of the ubiquitous concept of a group, called a "pregroup". In retrospect, this approach was already implicit in the work of Zellig Harris, Chomsky's teacher, and may be viewed as a refinement of the latter.

Lambek:"Word to Sentence"
0. Preface.

- My own experience tells me that, in countries where high school students are faced with a choice between a scientific program and a humanistic one based on a study of classical languages, future mathematicians often arise from the latter stream.
- The principal idea is to assign to each word in the mental dictionary one or more types. (Sometimes the types of inflected forms of lexical items may be calculated by certain "metarules".) These types are elements of an algebraic system or terms of a logical system and the grammatical status of a string of words is determined by a calculation on the corresponding types.

Lambek:"Word to Sentence"

36. Postscript on Minimalism.

- Readers familiar with current linguistic theory will have noticed that many ideas underlying the present monograph have been influenced by the profound insights gained by Chomsky and his school in the last half century. However, there are technical differences in how these ideas are displayed. Mainstream (Chomskyan) grammarians assume that words are arranged in labeled bracketed strings, or equivalently as leaves on a plane tree with labeled nodes, and that sentences are generated and analyzed by manipulating these structures.

- In categorial grammars, such calculations are performed not on labeled bracketed strings of words, but on strings of types, and this is assumed to be done in real time, even as the words are uttered or perceived during actual communication. In particular, this approach rules out any movement from the end of a sentence to the beginning.
- Chomsky's ideas have undergone a number of significant changes over the years, which have usually led his many disciples to revise their methodology as well. (I am reminded of the French comic book "Asterix and Cleopatra", in which Obelix accidentally breaks off the nose of the Sphinx, whereupon the souvenir vendors hasten to hack off the noses of their replicas.)

- For the last dozen years or so, the official version of the Chomskyan enterprise has been the so-called "minimalist program" (described as "maximalist" by some). It is not clear to me which of the earlier notions have been discarded and which have been absorbed in the new program. The two texts I have looked into rely heavily on a discussion of the earlier theories. For example, Chomsky [1995] discusses government theory, binding theory, case theory, *X*-bar theory, phrase structure theory and theta-theory, to judge by the section headings only, not counting trace theory and optimality theory, which are mentioned elsewhere (see Lasnik et al.[2005]).
  Is the reader required to study all these theories before engaging on the minimalist program?
  I must confess that I haven't mastered all these theories, although I suspect that what some of them are about can be incorporated into the pregroup framework.

- Much of the recent debate among mainstream grammarians is couched in the technical language of parsing trees. There have been a number of papers analyzing the minimalist program from an algebraic point of view, for example by Stabler [1997], and by several authors in de Groote et al. [2001], showing that minimalist grammars are weakly equivalent to multiply context-free ones. As far as I can judge, they still rely on an arboreal representation. While I cannot say that I have completely understood the minimalist program, I believe that in spirit, if not in technical details, pregroup grammars may be viewed as an attempt to carry out such a program.

文法の階層性
Chomsky Hierarchyについて

Chomsky Hierarchy

- 形式的言語の形式的文法は、次のような階層をなすことが知られている。これをChomsky Hierarchyと呼ぶ。
  - タイプ-0 文法は、全ての形式文法を包含する。
  - タイプ-1 文法は、文脈依存言語を生成する。
  - タイプ-2 文法は、文脈自由言語を生成する。
  - タイプ-3 文法は、正規言語を生成する。
- 機械が、この階層の中に位置付けられるどの形式的な文法を理解・学習できたかを考えることができる。

https://goo.gl/lR960c

Chomsky Hierarchy

Grammar	Languages	Automaton	Production rules (constraints)
Type-0	Recursively enumerable	Turing machine	$\alpha \rightarrow \beta$ (no restrictions)
Type-1	Context-sensitive	Linear-bounded non-deterministic Turing machine	$\alpha A \beta \rightarrow \alpha \gamma \beta$
Type-2	Context-free	Non-deterministic pushdown automaton	$A \rightarrow \gamma$
Type-3	Regular	Finite state automaton	$A \rightarrow a$ and $A \rightarrow aB$

https://en.wikipedia.org/wiki/Chomsky_hierarchy

Chomsky Hierarchyと Deep Learningでの文法理解の取り組み

- 機械が、この階層の中に位置付けられるどの形式的な文法を理解・学習できたかを考えることができる。
- Hochreiterの仕事は、機械が、有限オートマトンで表現される正規文法を、理解・学習できることを示唆している。
- Karpathyの仕事は、機械が、BNF(Backus-Naur Form)で表現されるような、Latex, プログラム言語の文脈自由文法を、理解・学習できることを、強く示唆している。

Chomsky Hierarchyと Deep Learningでの文法理解の取り組み

	文法	例	Deep Learning
Type-0	帰納的可算	チューリングマシン	
Type-1	文脈依存文法	自然言語*	?
Type-2	文脈自由文法	プログラム言語	Karpathy **2015年**
Type-3	正規文法	有限オートマトン	Hochreiter **1997年**

自然言語の多くの文法は、Contex-Freeで記述できる。
Mildly Context Sensitive Languages',proposed by Aravind Joshi.

Chomsky Hierarchyと Deep Learningでの文法理解の取り組み

	文法	例	Deep Learning
Type-0	帰納的可算	チューリングマシン	
Type-1	文脈依存文法	自然言語	Google Neural Machine Translation System ？ **2016年**
Type-2	文脈自由文法	プログラム言語	Karpathy **2015年**
Type-3	正規文法	有限オートマトン	Hochreiter **1997年**

自然言語の文法は、この階層のどこに位置付けられるか？

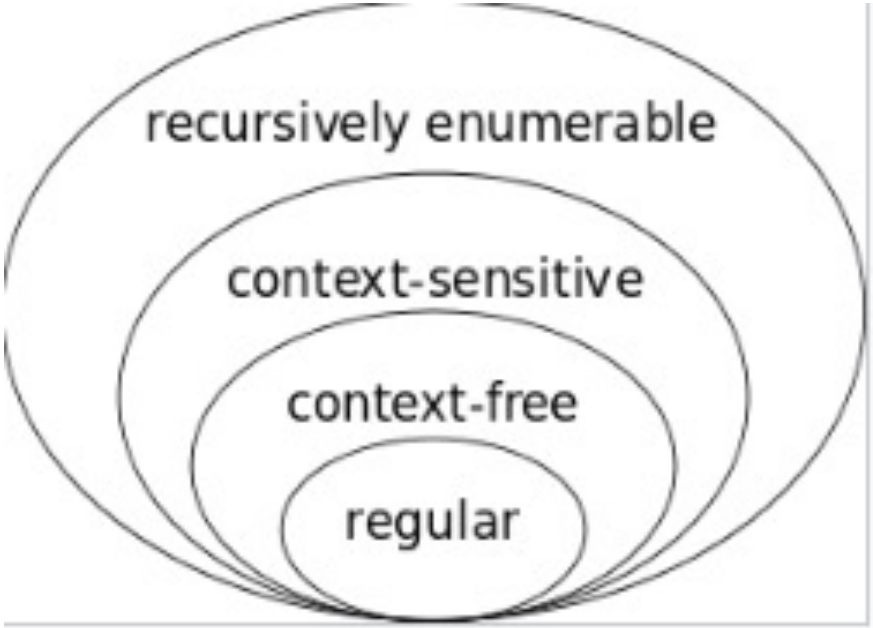
- 問題は、自然言語の文法が、この階層のどこに位置付けられるかということである。

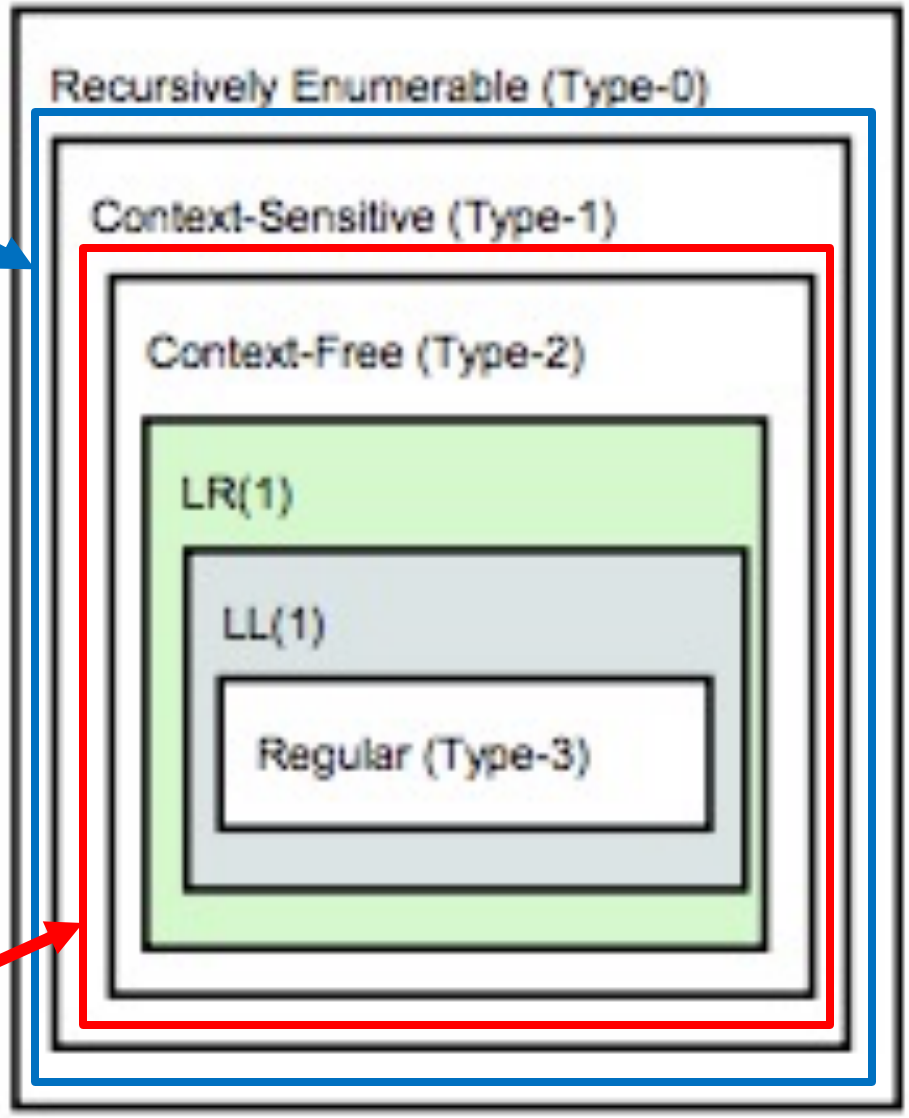
- 元の4階層の区分は、基本的だが粗いものである。Context-Freeな文法の中にも、LRやLLがあるように。
- かつてのChomskyは、英語は Regular 文法ではないと明確に述べたが、それが Context-Freeであるかどうかは、明確には判断を下していないようにも見える。ただ、自然言語は、Context-Free性を満たさないというのが、大方の意見のように見える。それは、Context-Freeを超えた、Mildly Context-Sensitive な文法だという。（次の図の、赤い四角の部分）

Minimalist ProgramとMerge

- 現在のChomskyのMinimalist Programでは、自然言語の文法を、文法の階層の中に位置付けようとする問題意識は、あまりないようにも見える。
- Category文法の人たちは、Minimalist文法が、weakly equivalent to multiply context-free ones だと言っているようだ。
- ただ、Minimalist Programの中心的概念であるMergeについては、それがRecursiveであることが強調されている。Recursiveというのは、Recursive Enumerableの一部分で、決定可能なものである。（図の青い四角の部分）
- 人間の言語能力と人間の数学的能力の違いを考えるとき、RecursiveとRecursive Enumerableの違いは、大きな意味を持つ。

Recursive Language
“Merge” is recursive





自然言語？

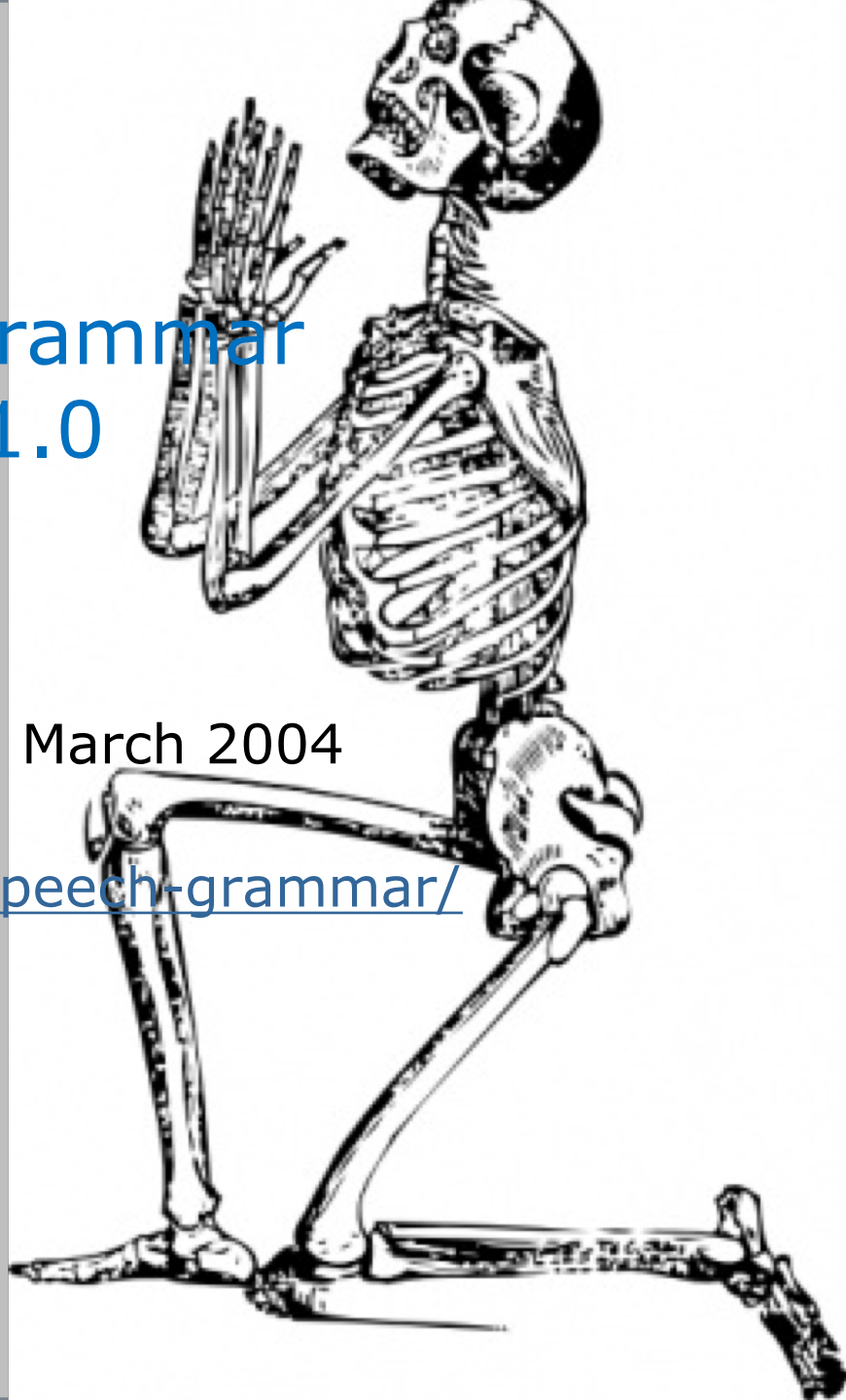
Mildly Context Sensitive Languages

Speech Recognition Grammar Specification Version 1.0

W3C Recommendation 16 March 2004

https://www.w3.org/TR/speech-grammar/

ここでの「文法」は、正規文法でしかない。
それでは、コンピューター言語もコンパイラーも、作れない。



Proof. Omitted. □

Lemma 0.1. *Let $\mathcal{C}$ be a set of the construction.*

Let $\mathcal{C}$ be a gerber covering. Let $\mathcal{F}$ be a quasi-coherent sheaves of $\mathcal{O}$-modules. We have to show that

$$\mathcal{O}_{\mathcal{O}_X} = \mathcal{O}_X(\mathcal{L})$$

.

Proof. This is an algebraic space with the composition of sheaves $\mathcal{F}$ on $X_{\acute{e}tale}$ we have

$$\mathcal{O}_X(\mathcal{F}) = \{morph_1 \times_{\mathcal{O}_X} (\mathcal{G}, \mathcal{F})\}$$

where $\mathcal{G}$ defines an isomorphism $\mathcal{F} \to \mathcal{F}$ of $\mathcal{O}$-modules. □

Lemma 0.2. *This is an integer $\mathcal{Z}$ is injective.*

Proof. See Spaces, Lemma ??. □

Lemma 0.3. *Let S be a scheme. Let X be a scheme and X is an affine open covering. Let $\mathcal{U} \subset \mathcal{X}$ be a canonical and locally of finite type. Let X be a scheme. Let X be a scheme which is equal to the formal complex.*

The following to the construction of the lemma follows.

Let X be a scheme. Let X be a scheme covering. Let

$$b : X \to Y' \to Y \to Y \to Y' \times_X Y \to X.$$

be a morphism of algebraic spaces over S and Y.

Proof. Let X be a nonzero scheme of X. Let X be an algebraic space. Let $\mathcal{F}$ be a quasi-coherent sheaf of $\mathcal{O}_X$-modules. The following are equivalent

(1) $\mathcal{F}$ is an algebraic space over S.
(2) If X is an affine open covering.

Consider a common structure on X and X the functor $\mathcal{O}_X(U)$ which is locally of finite type. □

Karpathy
http://goo.gl/mNqwCv

RNNが産み出した
数学論文モドキ

Stack Theory の教科書を
「学習」させたもの

This since $\mathcal{F} \in \mathcal{F}$ and $x \in \mathcal{G}$ the diagram

$$\begin{array}{ccc} \mathcal{S} & \longrightarrow & \\ \downarrow & & \\ \xi & \longrightarrow & \mathcal{O}_{X'} \\ & & \uparrow \\ \text{gor}_s & & \end{array} \qquad \begin{array}{ccc} & & \\ & & \\ & \searrow & \\ & & \end{array}$$

$$\begin{array}{ccc} = \alpha' & \longrightarrow & \\ \updownarrow & & \\ = \alpha' & \longrightarrow & \alpha \end{array} \qquad \begin{array}{c} X \\ \downarrow \\ \end{array}$$

$$\text{Spec}(K_\psi) \qquad \text{Mor}_{Sets} \quad \text{d}(\mathcal{O}_{\mathcal{X}_{X/k}}, \mathcal{G})$$

is a limit. Then $\mathcal{G}$ is a finite type and assume S is a flat and $\mathcal{F}$ and $\mathcal{G}$ is a finite type f_*. This is of finite type diagrams, and

- the composition of $\mathcal{G}$ is a regular sequence,
- $\mathcal{O}_{X'}$ is a sheaf of rings.

Proof. We have see that $X = \text{Spec}(R)$ and $\mathcal{F}$ is a finite type representable b algebraic space. The property $\mathcal{F}$ is a finite morphism of algebraic stacks. Then th cohomology of X is an open neighbourhood of U.

Proof. This is clear that $\mathcal{G}$ is a finite presentation, see Lemmas ??.
A *reduced above* we conclude that U is an open covering of $\mathcal{C}$. The functor $\mathcal{F}$ is a "field

$$\mathcal{O}_{X,x} \longrightarrow \mathcal{F}_{\overline{x}} \quad \text{-1}(\mathcal{O}_{X_{\acute{e}tale}}) \longrightarrow \mathcal{O}_{X_\ell}^{-1}\mathcal{O}_{X_\lambda}(\mathcal{O}_{X_\eta}^{\overline{v}})$$

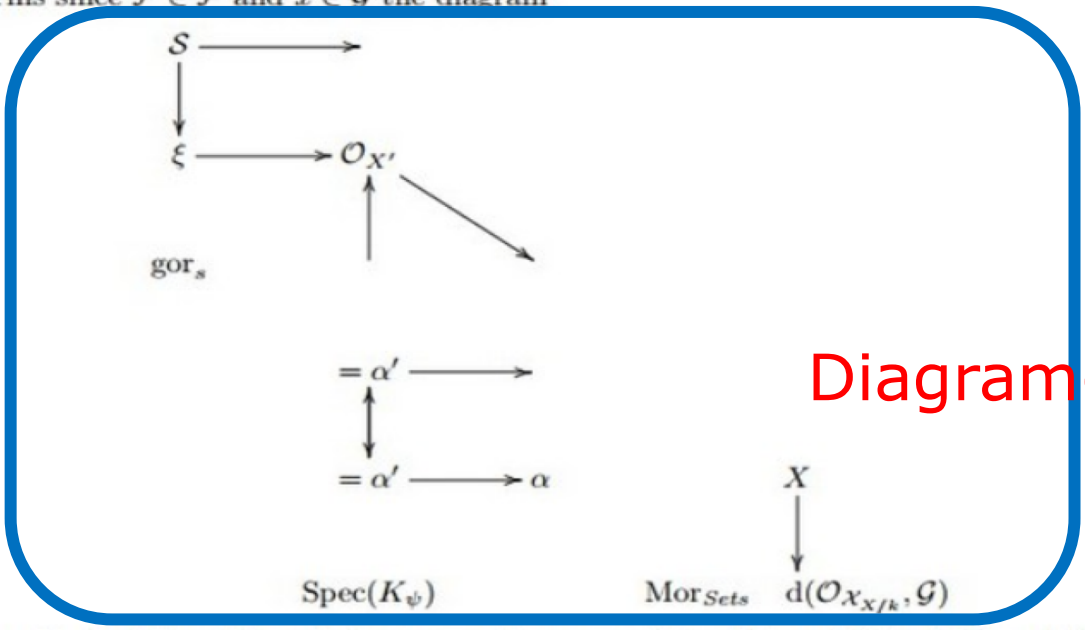
is an isomorphism of covering of $\mathcal{O}_{X_i}$. If $\mathcal{F}$ is the unique element of $\mathcal{F}$ such that X is an isomorphism.
The property $\mathcal{F}$ is a disjoint union of Proposition ?? and we can filtered set of presentations of a scheme $\mathcal{O}_X$-algebra with $\mathcal{F}$ are opens of finite type over S.
If $\mathcal{F}$ is a scheme theoretic image points. □

If $\mathcal{F}$ is a finite direct sum $\mathcal{O}_{X_\lambda}$ is a closed immersion, see Lemma ??. This is a sequence of $\mathcal{F}$ is a similar morphism.

RNNが産み出した数学論文モドキ

Stack Theory の教科書を「学習」させたもの

This since $\mathcal{F} \in \mathcal{F}$ and $x \in \mathcal{G}$ the diagram



is a limit. Then $\mathcal{G}$ is a finite type and assume S is a flat and $\mathcal{F}$ and $\mathcal{G}$ is a finite type f_*. This is of finite type diagrams, and

- the composition of $\mathcal{G}$ is a regular sequence,
- $\mathcal{O}_{X'}$ is a sheaf of rings.

Proof. We have see that $X = \mathrm{Spec}(R)$ and $\mathcal{F}$ is a finite type representable b algebraic space. The property $\mathcal{F}$ is a finite morphism of algebraic stacks. Then th cohomology of X is an open neighbourhood of U.

Proof. This is clear that $\mathcal{G}$ is a finite presentation, see Lemmas ??.
A *reduced above* we conclude that U is an open covering of $\mathcal{C}$. The functor $\mathcal{F}$ is a "field

$$\mathcal{O}_{X,x} \longrightarrow \mathcal{F}_{\overline{x}} \quad \text{-1}(\mathcal{O}_{X_{\acute{e}tale}}) \longrightarrow \mathcal{O}_{X_\ell}^{-1}\mathcal{O}_{X_\lambda}(\mathcal{O}_{X_\eta}^{\overline{v}})$$

is an isomorphism of covering of $\mathcal{O}_{X_i}$. If $\mathcal{F}$ is the unique element of $\mathcal{F}$ such that X is an isomorphism.
The property $\mathcal{F}$ is a disjoint union of Proposition ?? and we can filtered set of presentations of a scheme $\mathcal{O}_X$-algebra with $\mathcal{F}$ are opens of finite type over S.
If $\mathcal{F}$ is a scheme theoretic image points. □

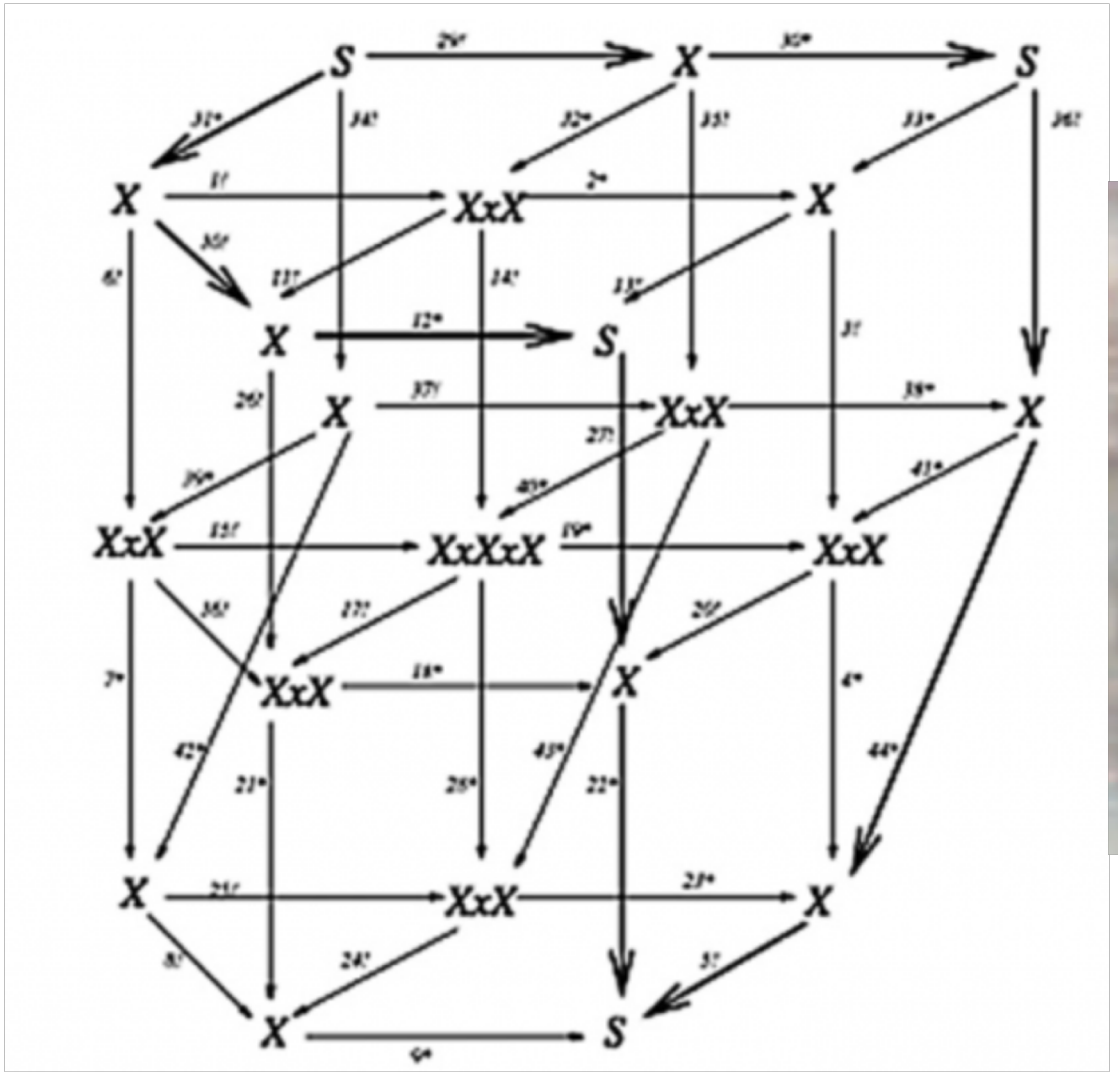
If $\mathcal{F}$ is a finite direct sum $\mathcal{O}_{X_\lambda}$ is a closed immersion, see Lemma ??. This is a sequence of $\mathcal{F}$ is a similar morphism.

Diagramの真似ができない！

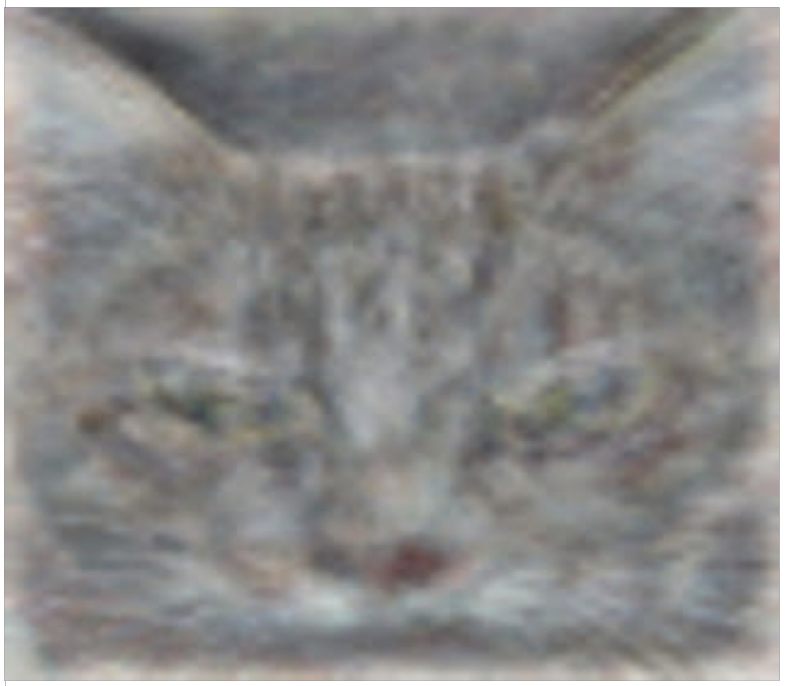
RNNが産み出した数学論文モドキ

Stack Theory の教科書を「学習」させたもの

生成できない！



生成できる！



先の空白部分には、
こんな図形が入るはず



の言語像

https://goo.gl/ny5rSM

普遍文法をボトムアップで構成する

"Approaching UG from Below"

Noam Chomsky 2007

http://goo.gl/V7lrsG

言語能力は「認識の器官」
普遍文法は、言語能力の初期状態

- 50年前に形を取り始めた「生物言語学（biolinguistic）の展望」の中で、主要な関心は、言語能力（Language Faculty) FLへの遺伝子の寄与を決定する努力へと変わっている。言語能力 FLは、「認識の器官」であると理解され、その点では、仮想的にはすべての人間に共有された、本質的に重要な点で人間のみにユニークなものである。それゆえ、人間という種のある種の財産である。
  そのように解釈されるなら、言語は、FLの一つの状態である **I-Language**であり、普遍文法（UG）は、FLの初期状態の理論と、再解釈される。

言語発達の三つの要因

- あきらかに、個人における言語発達には、次の三つのファクターが含まれていなければならない。

1. 遺伝子の寄与。これは達成可能な言語に、幾つかの制限を課し、それゆえ、言語獲得を可能にする。
2. 外的なデータ。狭い範囲の中で、一つか他の言語を選択する経験へと変換される。
3. 言語能力 FL に特有ではない、いくつかの原理。

"Three Factors in Language Design"

http://goo.gl/IJo7Ru Chomsky 2005年

- 第三のファクターは、いくつかのサブタイプに分かれる。
  (a) データ分析の原理。これは、言語獲得とその他の領域で利用されるだろう。
  (b) 構造的なアーキテクチャーと発達上の制約の原理。これには、機能の連携、器官の形成、広い範囲に及ぶ活動が含まれる。また、効率的な計算の原理も含まれる。それは、言語のような計算システムにとって特別の重要性があると期待される。
  これらのサブタイプのうちの、第二のものが、達成可能な言語の本性を決定するのに特別な重要性を持つべきである。

「トップダウン」と「ボトムアップ」
ミニマリスト・プログラムは、ボトムアップ

- 生成文法の現代の歴史を通じて、言語能力の特性を決定する問題は、「トップダウン」でアプローチされてきた。例えば、言語獲得を説明するためには、普遍文法には、どれだけの属性を持たねばならないのか？　といったように。
- ミニマリスト・プログラムは、「ボトムアップ」からのアプローチを探求する。第三のファクターに依拠して、達成されたI-Languageの多様性を説明しながら、いかに少数の属性に普遍文法に帰することができるのか？　と。
- 二つのアプローチは、もちろん、収束しなければならない。また、共通の目的の追求の中で、相互作用しなければならない。

I-languageと二つのインターフェース

- I-language は、無限に多くの内的表現を生成する計算システムである。その表現のそれぞれが、感覚運動的と概念・思考的の二つのインターフェース・システムへの命令の配列とみなすことができる。
- 第三のファクターの条件が機能するところまで、言語は、インターフェースで課せられた条件を満たすように、効率的にデザインされるであろう。もっとラディカルなテーゼを想像することができるのだが、それについては、後で簡単に帰ってくるだろう。

Merge

- もっとも初等的な形式では、生成システムは、すでに形成された構造たちを取りだし、一つの新しい構造に結合する操作に基礎を置く。それを”Merge”と呼ぼう。制限なしで働いて、Mergeは、別々の無限の構造化された表現を生み出す。それゆえ、Mergeと、それが制限なしで適用されるという条件も、普遍文法の一部となる。

Mergeとは何か？

“The Science of Language: Interviews with James McGilvray” から
https://goo.gl/lbAO7n

Mergeとは何か？

人間の進化史の中で

- ある人の上に、その子孫にずっと伝えられる何かが起きた。あきらかに非常に短い間に、その遺伝子の変化はそのグループ内で優勢になった。それは、淘汰上の何らかの有利さがあったに違いない。ただ、それは非常に短い時間に、血縁上の小さなグループで起きた。さて、それは、なんだったのだろうか？ もっとも単純な想定は、それを疑う理由は何もないのだが、我々に起きたこととは、我々は Mergeを獲得したということだ。人は、すでに構成された心の中のオブジェクトをとって、それから、さらに大きな心の中のオブジェクトを構成することを可能にする操作を獲得したのだ。それがMergeだ。それを獲得するやいなや、人は、利用可能な表現や思考の階層的構造の無限の多様性をもつことになる。

Mergeとは何か？

感覚・運動システムと思考システムの先在

- Mergeが導入された時、我々は、すでに感覚・運動システムを持っていた。おそらくそれは、周辺的なものとして利用された。事実、それを外部化しようという考えは、もっとあとになってうまれたに違いない。
  さらに、我々は、ある種の思考のシステムを持っていた。しかし、それは原始的なもので、物事をなんらかのしかたで心に思い描くといったようなものだった。たとえそれがどんなものであれ、それらは動物のシステムと似たものには見えない。それが、17,18世紀によく議論された理由だ。ただ、あきらかに、それは存在していた。

Mergeとは何か？

人間の思考と言語の誕生

- ひとたび、人がこの構成の技術と、これら（Chomskyが、「概念・思考システム」と呼ぶ思考システム）を使った、階層的に構造化された表現の無限の多様性を獲得すると、人は、突然、それまで誰もできなかったやり方で、考え、計画し、解釈することができるようになった。…
  そして、この多様な思考を、なんとかして外部化しようという考えが生まれる。… そうしたものとして、言語の進化を捉えることができる。

Mergeとは何か？
“Minimalist program” wiki Page

ここでは、Mergeとはどういうものであるのか、簡単にな例に基づいて述べる。

https://goo.gl/x7Cyx6

Merge

- Merge は、二つのオブジェクト（αとβとしよう）をとって、ラベルを持った（αかβである。この例では、αがラベルになっている）一つの順序を持たない集合に併合する関数である。ラベルは、この句（phrase）の性質を表す。

Merge (α, β) → {α, {α, β}}

- 例えば、Mergeは、Lexical Item “drink”と”water” に作用して、”drink water”を与える。

句 “drink water” の性質

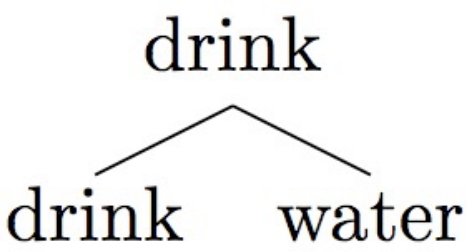
- この句 “drink water”は、名詞water のようにではなく、動詞drinkのように振る舞うことに注意すること。
- 次の例のように、drinkが置ける位置に、drink waterを置くことができる。
  - I like to _____(drink)/(drink water).
  - (Drinking/Drinking water) ____is fun.
- さらに、句drink waterは、名詞water と同じ場所には置くことができない
  - There's some *water* on the table
  - *There's some *drink water* on the table

ラベルの選択
Merge (drink, water) の場合

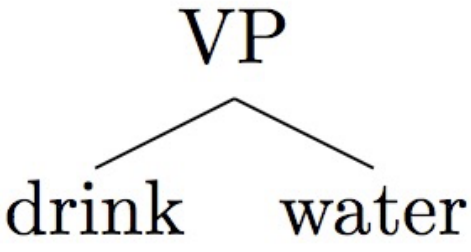
- 句は、ラベルと同じものとみなされる。”drink water”の例では、この句は動詞として振舞うので、ラベルは、drinkになる。

  Merge (drink, water)
  → {drink, {drink, water}}

- この句は、動詞句（VP: Verb Phrase）とよばれる。構文木で表すと、次のように表現できる。



あるいは



ラベルの選択

Merge (cool, water) の場合

- 今度は、”cool”と”water”をMergeすることを考えよう。この時、この句 “cool water”は、動詞drink のようにではなく、名詞waterのように振る舞うことがわかる。
- 次の例のように、waterが置ける位置に、cool waterを置くことができる。
  - I like to drink _____ (water)/(cool water).
  - Drinking _____ (water)/(cool water) is fun.
- ”cool water”のラベルはwaterで、名詞句（NP）である。

Merge (cool, water)
→ {water, {cool, water}}

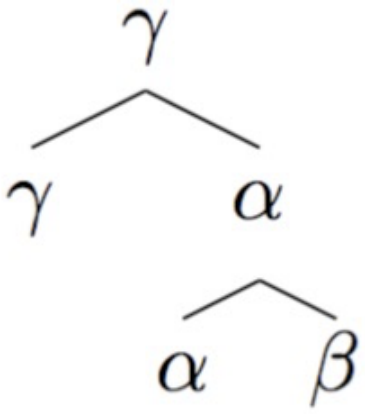
既に構成された句とのMerge

- Mergeは、既に構成されたものにも適用できる。それができなかったら、こうしたシステムは、文法的には、二語文しか生成できないだろう。もし新しいhead γが、既に構成されているオブジェクト（句だ）とMergeされれば、次のようになる。

  Merge (γ, {α, {α, β}})

  → {γ, {γ, {α, {α, β}}}}

- 次のような構文木になる。
  ここで、γ はラベルである。



Language = Interfaces + Recursion

計算システムとしての言語能力

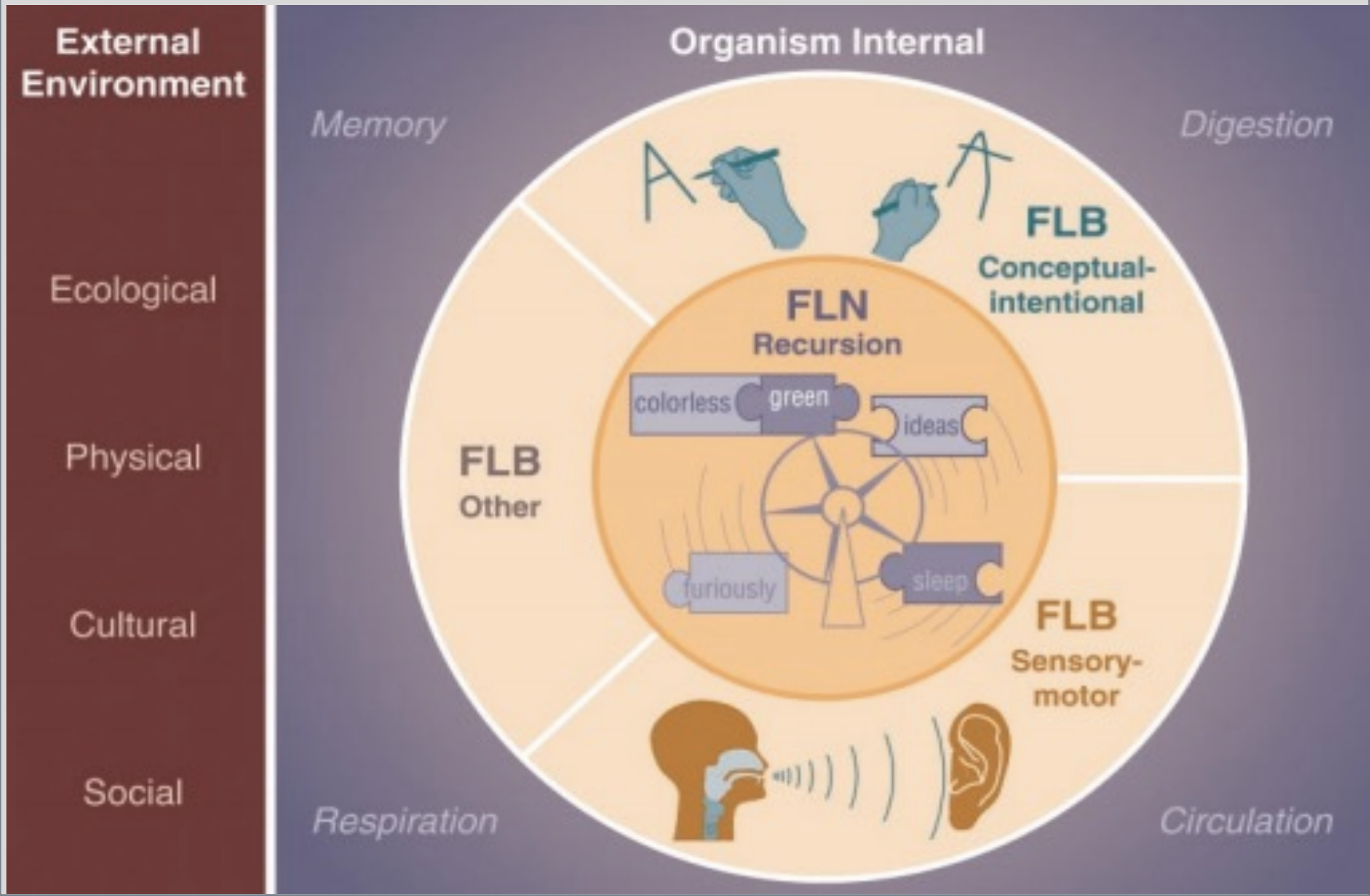


https://goo.gl/i0SVEP

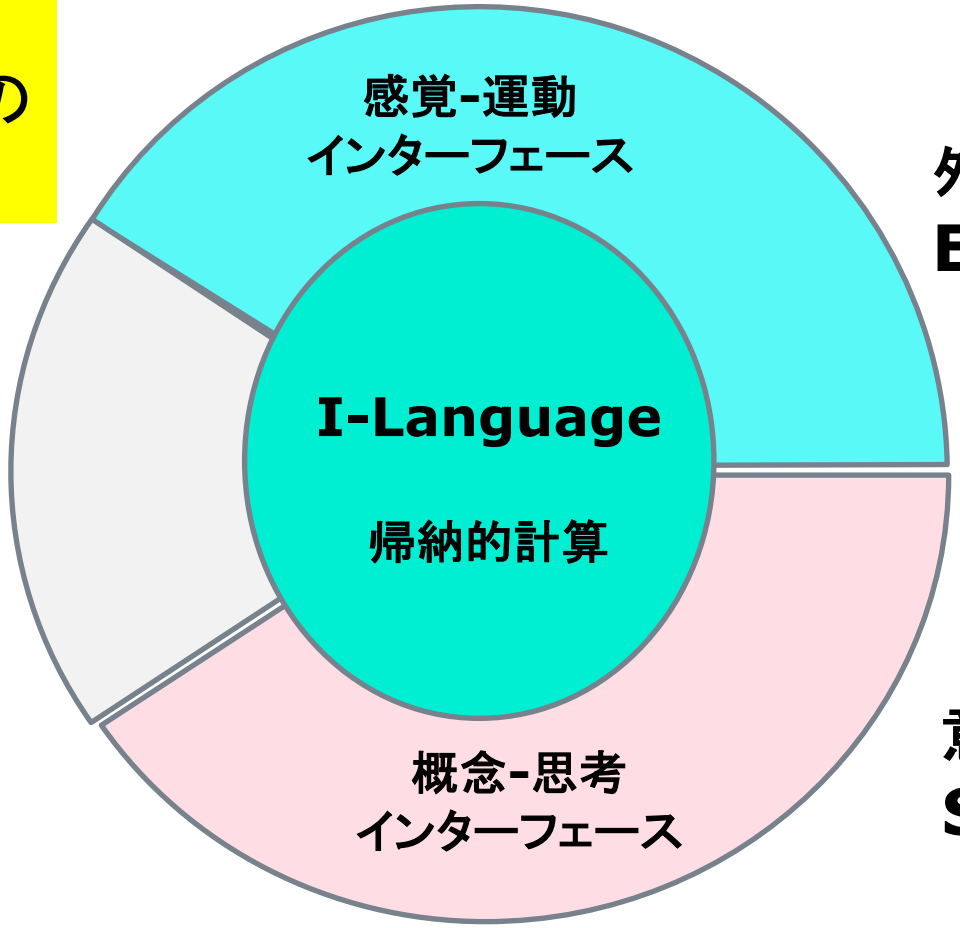
「計算システム」としての言語能力

- Minimalist Programでは、言語能力を、二つのインターフェースをもつ計算システムとして捉える。
- こうしたアプローチを、ボトムアップから積み上げてゆくということになれば、それは、機械による言語能力の実装に、近ずいてゆくことになるだろう。
- ただ、そうした実装を、僕は知らない。

"The Faculty of Language: What Is It, Who Has It, and How Did It Evolve?" http://goo.gl/pgSppH



Minimalist Programでの言語モデル

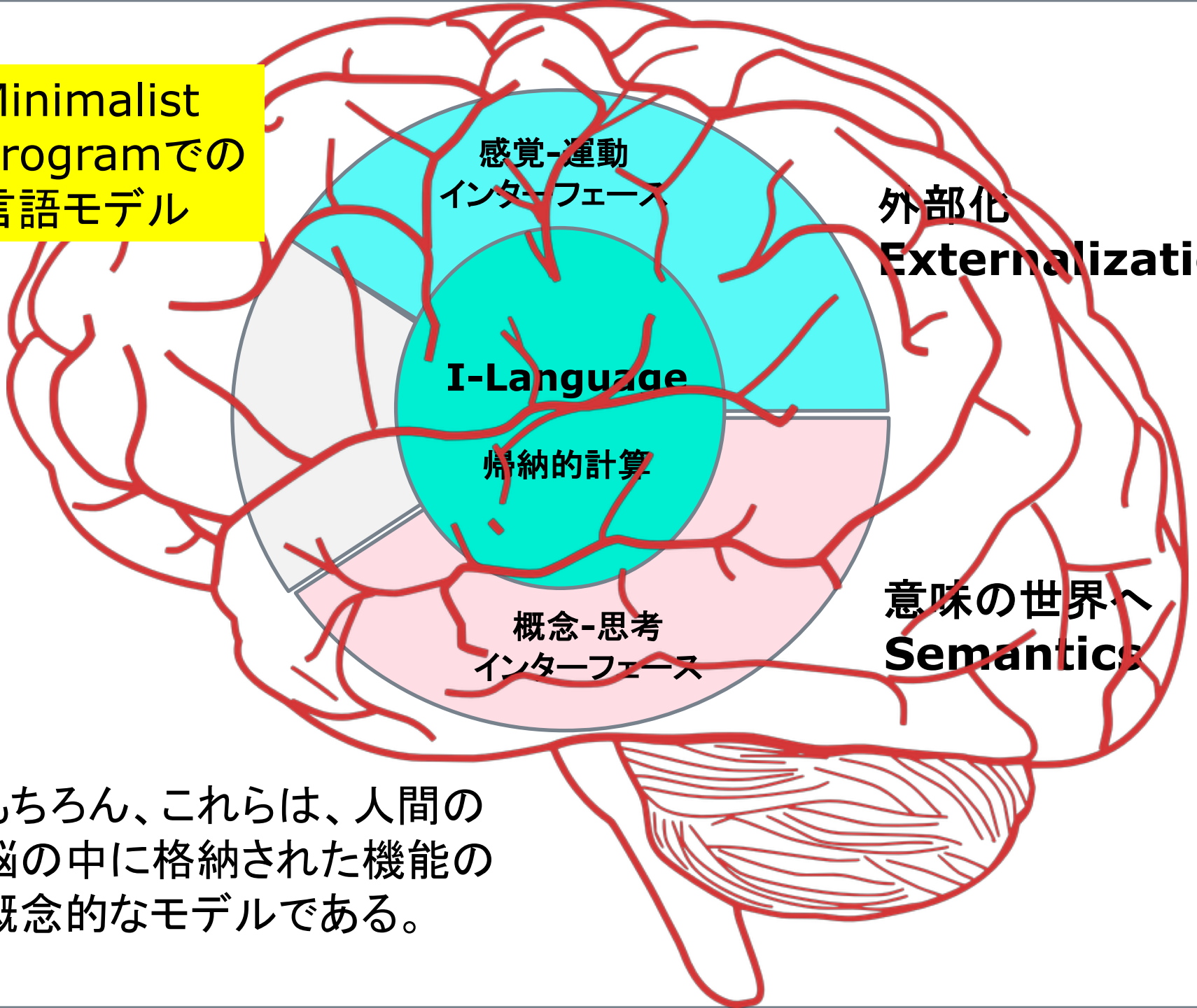


外部化
Externalization

意味の世界へ
Semantics

Language = Interfaces + Recursion

Minimalist Programでの言語モデル



もちろん、これらは、人間の脳の中に格納された機能の概念的なモデルである。

Minimalist Programでの言語モデル

重要なことは、特に概念-思考インターフェースの先に、言語能力を支える、「概念のリソース」とでもいうべきものが存在していることである。



感覚-運動
インターフェース

Phonetic form

外部化
Externalization

I-Language

帰納的計算

Logical Form

概念-思考
インターフェース

概念
Semantics





概念のリソースの存在

- その理由は、いろんなところで議論してきたし、ここでは、その問題は脇に置こう。ただ、もしそうなら、言語能力の遺伝的に決定された、さらなるコンポーネントが存在する（すなわち、その上で、言語能力が描く概念のリソースが存在する）。そして、言語の進化の研究が取り組むべき問題が存在する。（すなわち、言語以前の、人間にも利用可能な概念のリソースの進化についてだ）

CIインターフェースの先にあるもの

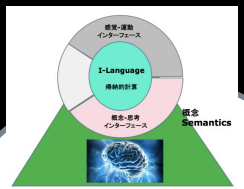
Bot

Botが行っていることは、言語モデル的には、こんなこと。

ただし、Bot内の小人の言語能力はフルに活用される。

もちろん、それはBotではなく、人間の能力である。





Text2Speech
Speech2Text



「中国人の部屋」と同じ構造をしている



計算システムを構成するもの

ここでは、言語の計算システムの概略を述べる。

言語表現（π, λ）については、論文 “Bare Phase Structure” https://goo.gl/LiFcAQ を参照のこと。

“Minimalist Program” の有名な **“Chapter 4”** は、このシステムのアルゴリズムの記述として読める！

lexical Item (語彙項目)

- Mergeが制限なしに適用可能なことに加えて、普遍文法は、少なくともアトミックな要素である語彙項目 Lexical Item LIを提供しなければならない。それは、それぞれがMergeや他の操作が表現を形成するのに適用する特性（特徴）の構造化された配列である。これらの諸特徴は、その並びが、インターフェースで解釈されるのに必要な情報を含んでいなければならない。すべての情報は、I-Languageと同じ範囲で、効率的な計算のための自然な条件であるInclusiveness Condition を満たす。

Inclusiveness Condition :The machinery of syntax (C_{HL}) does not introduce any new features not already contained in the lexical items.

Inclusiveness Conditionの批判
https://goo.gl/Z6HDRd

Lexicon（語彙集）

- 言語は、少なくとも、パラメーターの評価と普遍文法によって利用可能になる諸特徴の格納庫からの選択によって特定される。そして、LI(Lexcon 語彙集)の中のこれらの諸特徴の組み合わせのリストは、さらなる条件を満たしている。人間のLIは、両方のインターフェースで、他の動物たちのもつシンボル要素とは、本質的に異なっているという重要な証拠がある。

CI インターフェースでは、動物のコミュニケーション・システムでは一般的な特性に見える、こころからは独立な要素への関係が欠落している。それは、音韻論では当然と思われていることと似ている。もし、そうであるなら、人間の言語には、参照に似た関係はない。それ故、Frege, Peirce, Tarski, Carnap, Quine やその他の、また現代の「外部主義」の理論家のいう、技術的な意味での「意味論」は存在しない。

colourless green ideas sleep furiously

Principal & Parameter



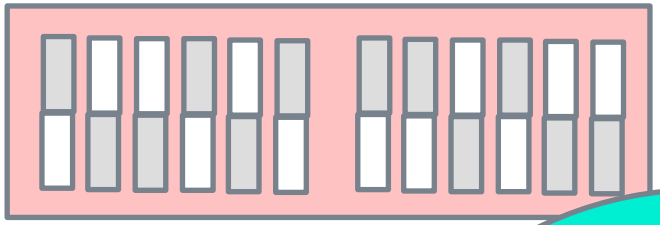
計算システムの
初期状態としての
普遍文法

パラメーターの
スイッチは入って
おらず、辞書も
空のままである。

Computing System

Lexicon

Principal & Parameter



後天的な経験で
パラメーターの
スイッチが入り、
文法は各国語化
する。

辞書も各国語の
語彙で満たされる。

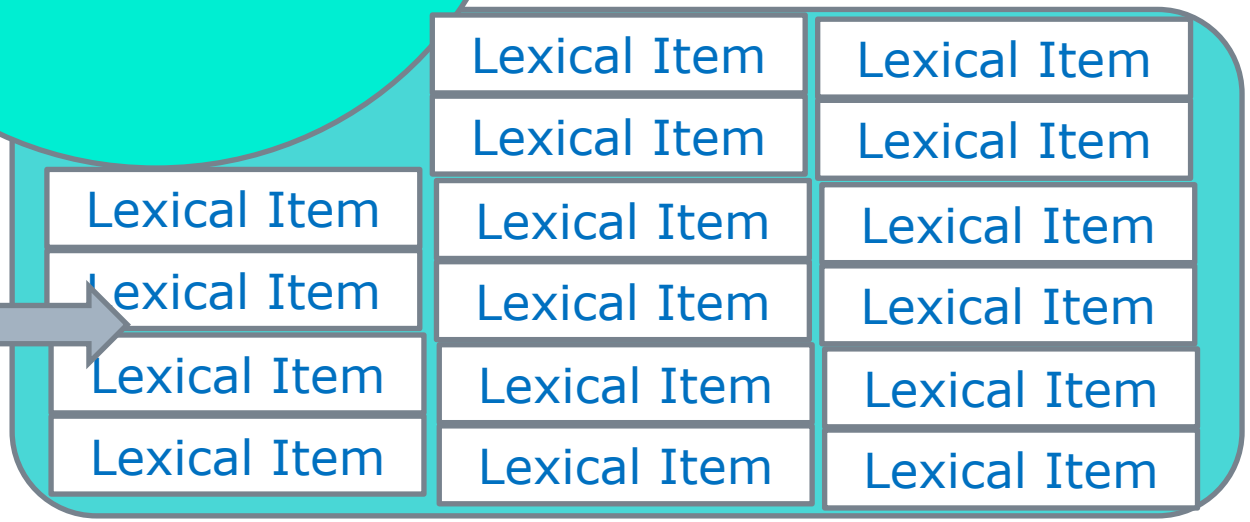
Computing System
Merge

Lexical Item

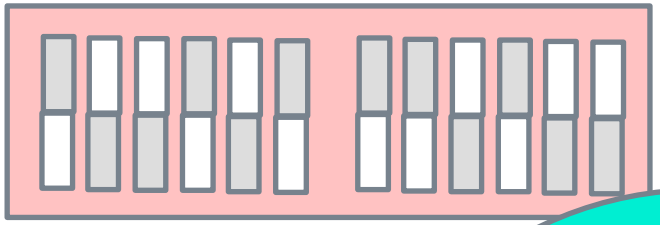
各語彙項目は、語彙記載項(lexical entry)の集合と語形成規則(word formation rule)などの規則の束から成る語彙目録(lexicon)によって、音韻・統語・意味などの情報が規定されている。

https://goo.gl/eJZMGv

Lexicon

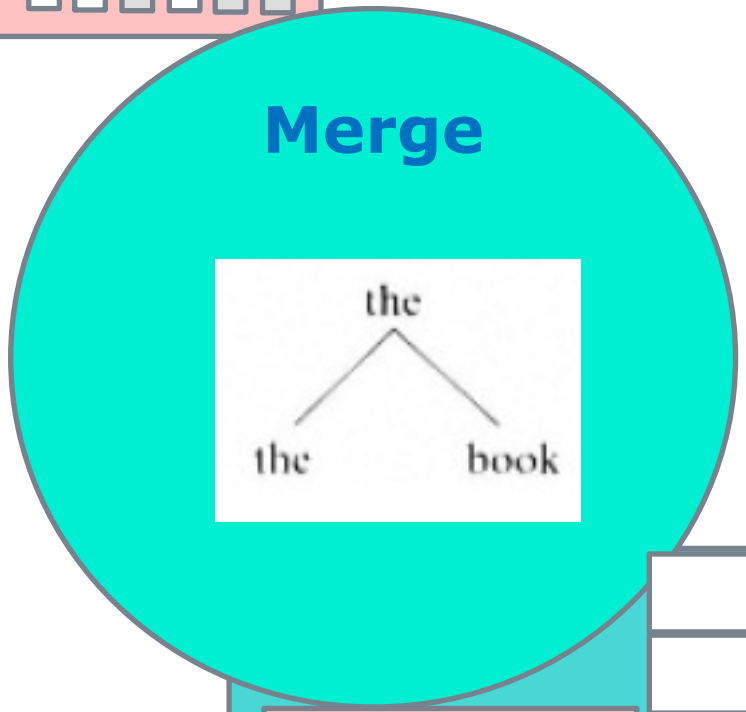


Principal & Parameter



Merge

Lexiconから、二つのLexical Item である“the”と”book”が取り出され、Mergeされたところ。

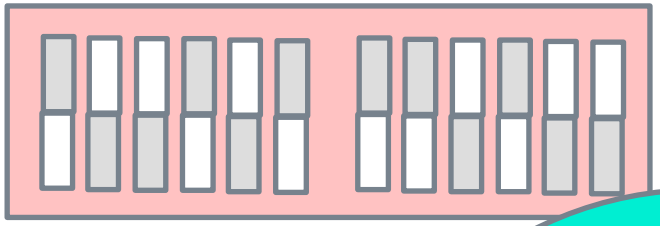


Lexicon

it	Lexical Item
girl	Lexical Item
mama	Lexical Item
see	Lexical Item
apple	Lexical Item
a	Lexical Item

the
book
man
eat

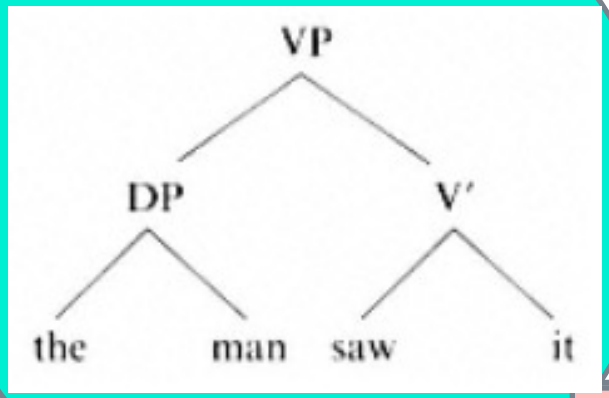
Principal & Parameter



Merge

Mergeの対象は、
Lexical Itemだけ
とは限らない。
既に形成された句も、
Mergeの対象になる。

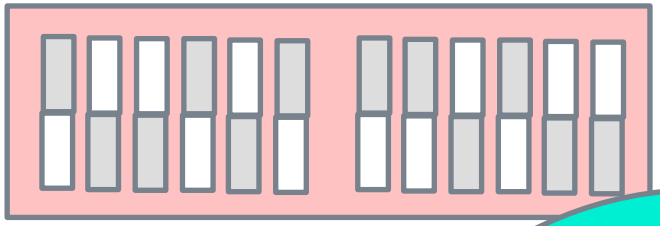
Merge



Lexicon

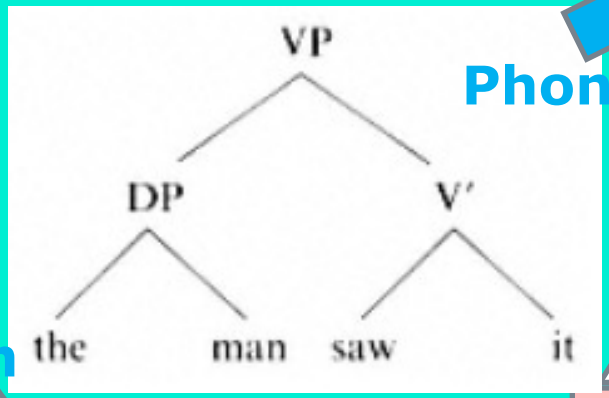
it	Lexical Item
girl	Lexical Item
the	Lexical Item
book	Lexical Item
man	Lexical Item
eat	Lexical Item
mama	Lexical Item
see	Lexical Item
apple	Lexical Item
a	Lexical Item

Principal & Parameter



SMインターフェースに対しては、Phonetic formが生成される。

Merge



Phonetic form

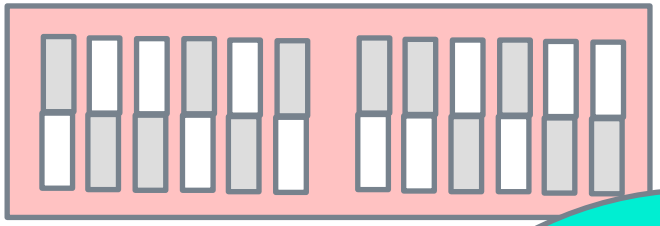
Logical Form

CIインターフェースに対しては、Logical formが生成される。

Lexicon

it	girl	Lexical Item
the	book	Lexical Item
man	eat	Lexical Item
mama	see	Lexical Item
apple	a	Lexical Item
Lexical Item	Lexical Item	Lexical Item

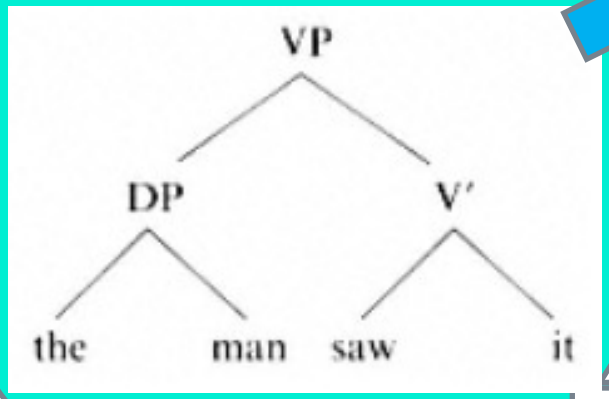
Principal & Parameter



SM Interface

Phonetic form

Merge



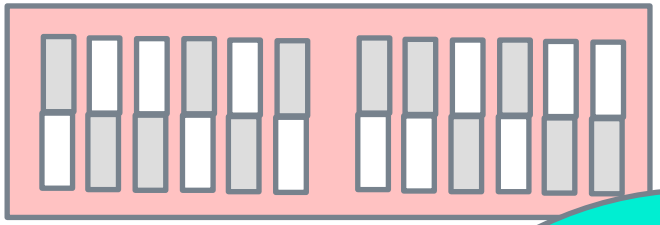
CI Interface

Logical Form

Lexicon

it		Lexical Item
girl		Lexical Item
the	mama	Lexical Item
book	see	Lexical Item
man	apple	Lexical Item
eat	a	Lexical Item

Principal & Parameter





SM Interface

Phonetic form

Merge

CI Interface

Logical Form



Lexicon

it		Lexical Item
girl		Lexical Item
the	mama	Lexical Item
book	see	Lexical Item
man	apple	Lexical Item
eat	a	Lexical Item

The mathematics of sentence structure

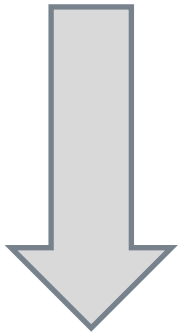
JOACHIM LAMBEK

http://lecomte.al.free.fr/ressources/PARIS8_LSL/Lambek.pdf

2. Syntactic types

$$\text{(I)} \quad (x/y)y \to x, \quad y(y\backslash x) \to x.$$

1958年の定式化



2008年の定式化

$$xx^r \to 1, \quad x^\ell x \to 1,$$

3. Type list for a fragment of English

(1)	John	works	
	n	$n\backslash s$	

n n\s → s

(2)	(poor	John)	works	
	n/n	n	$n\backslash s$	

(n/n n) n\s → n n\s → s

(3)	(John	works)	here	
	n	$n\backslash s$	$s\backslash s$	

(n n\s) s\s → s s\s → s

(4)	John	(never	works)	
	n	$(n\backslash s)/(n\backslash s)$	$n\backslash s$	

n ((n\s)/(n\s) n\s)
→ n n\s → s

(5)	(John	works)	(for	Jane)	
	n	$n\backslash s$	$(s\backslash s)/n$	n	

(n n\s) ((s\s)/n n))
→ s s\s → s

3. Type list for a fragment of English

(6)	(John	works)	(and	(Jane	rests))
	n	$n\backslash s$	$(s\backslash s)/s$	n	$n\backslash s$

(n n\s)) ((s\s)/s (n n\s))) → s ((s\s)/s s) → s s\s → s

(7)	John	(likes	Jane)
	n	$(n\backslash s)/n$	n

n ((n\s)/n n)
→ n n\s → s

(7')	(John	likes)	Jane
	n	$n\backslash(s/n)$	n

(n n\(s/n)) n)
→ s/n n → s

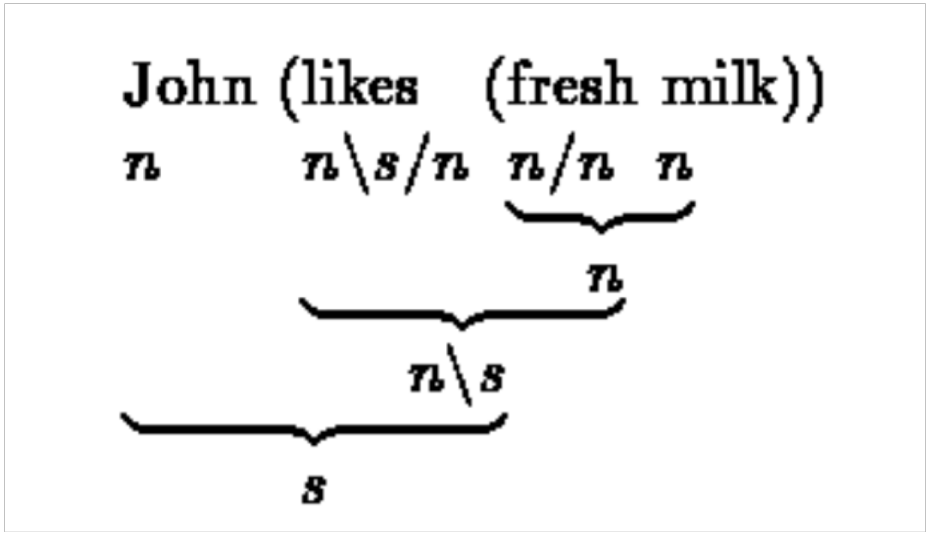
(II) $(x\backslash y)/z \longleftrightarrow x\backslash(y/z)$.

Table I

	Word	Type	Part of Speech	Fries Class
(1)	*works*	$n\backslash s$	intransitive verb	2C
(2)	*poor*	n/n	adjective	3
(3)	*here*	$s\backslash s$	adverb	4
(4)	*never*	$n\backslash s/(n\backslash s)$	adverb	
(5)	*for*	$s\backslash s/n$	preposition	F
(6)	*and*	$s\backslash s/s$	conjunction	E,J
(7)	*likes*	$n\backslash s/n$	transitive verb	2B

カテゴリー文法の「カテゴリー」は、数学のカテゴリー論の「カテゴリー」とは、直接の関係はない。それは、従来の文法の「品詞(Part of Speech)」概念の拡大である。その拡大された「品詞」をカテゴリーと呼んでいる。
このカテゴリーは、語の連接によって引き起こされる計算上の「型」を持つ。

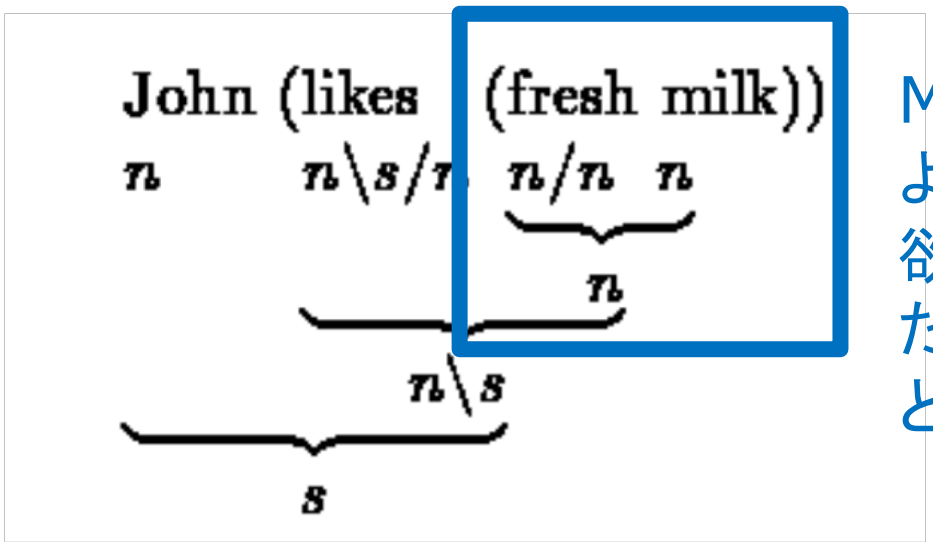
4. Formal systems



文の文法性の計算は、隣り合う語のカテゴリーの型の計算から、文のカテゴリー s を導出する形式的体系で記述できる。

$$n((n\backslash s/n)((n/n)n)) \rightarrow n((n\backslash s/n)n) \rightarrow n(n\backslash s) \rightarrow s.$$

4. Formal systems



Merge (cool, water)
→ {water, {cool, water}}
Mergeの議論での上のような導出を思い出して欲しい。同じことだ。
ただし、ここでは “water” というラベルは不要である。

$$n((n\backslash s/n)((n/n)n)) \rightarrow n((n\backslash s/n)n) \rightarrow n(n\backslash s) \rightarrow s.$$

5. Type computations in English

i. Insert brackets in all admissible ways.
ii. To each word assign all types permitted by a given finite type list. (We ignore for the moment the difficulty arising from words which possess a potentially infinite number of types, as do the chameleons `and' and `only').
iii. For each grouping and type assignment compute the type of the total expression.
iv. Select that method of grouping and that type assignment which yields the desired type s.

6. Pronouns

(8)	he	works,	he	likes	Jane
	$s/(n\backslash s)$	$n\backslash s$	$s/(n\backslash s)$	$n\backslash s/n$	n

(9)	that's	him,	Jane	likes	him,
	s/n	$(s/n)\backslash s$	n	$n\backslash s/n$	$(s/n)\backslash s$
	Jane	works	for	him	
	n	$n\backslash s$	$s\backslash s/n$	$(s/n)\backslash s$	

(10) he likes him ;
$s/(n\backslash s)$ $n\backslash s/n$ $(s/n)\backslash s$

$$(s/(n\backslash s))(n\backslash s/n)((s/n)\backslash s)$$

(III) $(x/y)(y/z) \rightarrow x/z, \quad (x\backslash y)(y\backslash z) \rightarrow x\backslash z.$

$$(s/(n\backslash s))(n\backslash s/n) \rightarrow s/n$$

$$(n\backslash s/n)((s/n)\backslash s) \rightarrow n\backslash s$$

$$\underbrace{(\text{he likes})}_{s/n} \quad \underbrace{\text{him},}_{(s/n)\backslash s} \quad \underbrace{\text{he}}_{s/(n\backslash s)} \quad \underbrace{(\text{likes him})}_{n\backslash s.}$$

(John	works)	for	Jane
n	$n\backslash s$	$s\backslash s/n$	n

$$(n\backslash s)((s\backslash s/n)n) \quad \rightarrow \quad (n\backslash s)(s\backslash s) \quad \rightarrow \quad n\backslash s.$$

$$n \quad \rightarrow \quad s/(n\backslash s), \qquad n \quad \rightarrow \quad (s/n)\backslash s$$

$$\text{(IV)} \quad x \rightarrow y/(x\backslash y), \quad x \rightarrow (y/x)\backslash y.$$

$$n^*(n^*\backslash n^*/n')n \rightarrow (n^*/n')n \rightarrow (n^*/n')n' \rightarrow n^*$$

From word to sentence: a computational algebraic approach to grammar

Joachim Lambek

http://www.math.mcgill.ca/barr/lambek/pdffiles/2008lambek.pdf

I. Introduction

1. Origin of languages.
2. Grammar and mathematics.
3. An algebraic approach to grammar.
4. Mathematical interlude.

basic types

- n = subject,
  n_3 = third person singular subject,
- $\mathbf{s}_1$ = declarative sentence in present tense,
- **i** = infinitive of intransitive verb,
- **j** = infinitive of any complete verb phrase,
- **o** = direct object.

- We postulate

  $n_3 \rightarrow n$, $\quad \mathbf{i} \rightarrow \mathbf{j}$,

partial order

- *reflexive*: $x \to x$,
- *transitive*: $x \to y$ and $y \to z$ implies $x \to y$, and
- *anti-symmetric*: $x \to y$ and $y \to x$ implies $x = y$

- We may read $x \to y$ as saying that everything of type x is also of type y.

$$(xy)z = x(yz)$$

$$1x = x = x1,$$

$$xx^r \to 1, \quad x^\ell x \to 1,$$

$$\begin{aligned}\pi_3(\pi^r \mathbf{s}_1 \mathbf{j}^\ell)(\mathbf{i}\mathbf{o}^\ell)\mathbf{o} &= [\pi_3\pi^r]\mathbf{s}_1\,[\mathbf{j}^\ell\mathbf{i}][\mathbf{o}^\ell\mathbf{o}] \\ &\to [\pi\pi^r]\mathbf{s}_1\,[\mathbf{j}^\ell\mathbf{j}]\,[\mathbf{o}^\ell\mathbf{o}] \\ &\to 1\mathbf{s}_1\,1\,1 \;=\; \mathbf{s}_1,\end{aligned}$$

she will come

$\pi_3\ (\pi^r \mathbf{s}_1 \mathbf{j}^l)\ \mathbf{i}$

In the above examples, the pronoun *she* has type π_3 to indicate that it serves as a third person singular subject.

It also has type π, since $\pi_3 \rightarrow \pi$. The instransitive verb *come* has type $\mathbf{i}$ and also $\mathbf{j}$, since $\mathbf{i} \rightarrow \mathbf{j}$.

$$\pi_3\ (\pi^r \mathbf{s}_1 \mathbf{j}^l)\mathbf{i} \rightarrow \pi\ (\pi^r \mathbf{s}_1 \mathbf{j}^l)\ \mathbf{i} \rightarrow \pi\ (\pi^r \mathbf{s}_1 \mathbf{j}^l)\ \mathbf{j}$$
$$\rightarrow \pi\ \pi^r \mathbf{s}_1 \mathbf{j}^l\ \mathbf{j} \rightarrow 1 \mathbf{s}_1 1 \rightarrow \mathbf{s}_1$$

she will see him

$\pi_3\ (\pi^r\mathbf{s}_1\mathbf{j}^l)(\mathbf{io}^l)\mathbf{o}$

The transitive verb *see* has type $\mathbf{io}^l$, to indicate that it is looking for an object on the right.
The superscript l is to show that $\mathbf{o}^l$ cancels $\mathbf{o}$ from the *left*, that is, $\mathbf{o}^l\mathbf{o} \to 1$.

The finite modal verb-form *will* here has type $\pi^r\mathbf{s}_1\mathbf{j}^l$ to indicate that it is looking for a subject on the left and for an infinitive on the right. The superscript r is to show that π^r cancels π from the *right*, hence also π_3, since

$$\pi_3\pi^r \to \pi\pi^r \to 1.$$

$$\pi_3\ (\pi^r\mathbf{s}_1\mathbf{j}^l)(\mathbf{io}^l)\mathbf{o} \to \pi\pi^r\mathbf{s}_1\mathbf{j}^l\mathbf{io}^l\mathbf{o} \to \pi\pi^r\mathbf{s}_1\mathbf{j}^l\,\mathbf{jo}^l\mathbf{o}$$
$$\to (\pi\pi^r)\mathbf{s}_1(\mathbf{j}^l\,\mathbf{j})(\mathbf{o}^l\mathbf{o}) \to 1\mathbf{s}_1 11 \to \mathbf{s}_1$$

she will see him

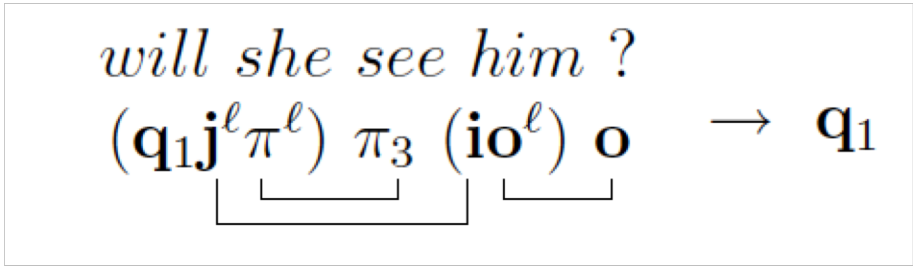
$\pi_3\ (\pi^r \mathbf{s}_1 \mathbf{j}^l)(\mathbf{io}^l)\mathbf{o}$

The superscript l again reminds us that $\mathbf{j}^l$ cancels $\mathbf{j}$ from the *left*, hence also $\mathbf{i}$, since

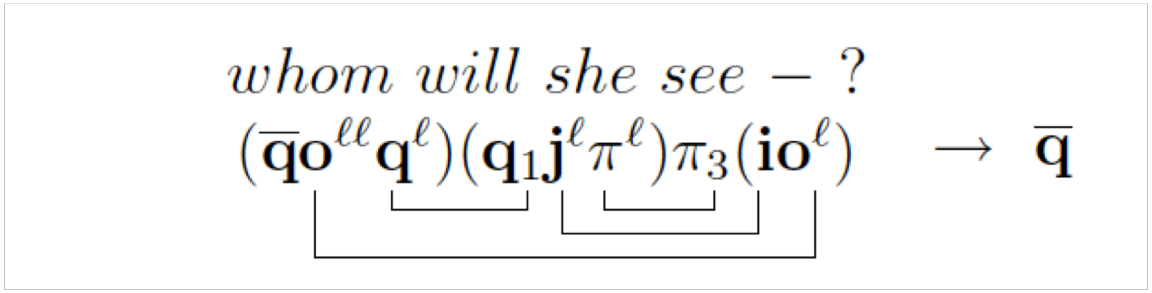
$$\mathbf{j}^l\mathbf{i} \to \mathbf{j}^l\mathbf{j} \to 1.$$

$$\begin{aligned}\pi_3(\pi^r \mathbf{s}_1 \mathbf{j}^\ell)(\mathbf{io}^\ell)\mathbf{o} &= [\pi_3\pi^r]\mathbf{s}_1\,[\mathbf{j}^\ell\mathbf{i}][\mathbf{o}^\ell\mathbf{o}] \\ &\to [\pi\pi^r]\mathbf{s}_1\,[\mathbf{j}^\ell\mathbf{j}]\,[\mathbf{o}^\ell\mathbf{o}] \\ &\to 1\mathbf{s}_1\,1\,1 \;=\; \mathbf{s}_1,\end{aligned}$$

With apologies to Chomsky, I would like to say that the parentheses (and) indicate the surface structure and the square brackets [and] the deep structure.



Here we have introduced a new basic type
$\mathbf{q}_1$ = yes or-no question in present tense
and assigned a new type $\mathbf{q}_1\mathbf{j}^l\pi^l$ to *will*.



where we have introduced two new basic types

$\mathbf{q}$ = yes-or-no question in any tense,

$\overline{\mathbf{q}}$ = arbitrary question (including wh-question)

and postulated

$$\mathbf{q}_1 \ \rightarrow\ \mathbf{q} \ \rightarrow\ \overline{\mathbf{q}}.$$

The distinction between q and q is necessary, since a question of type q can be preceded by when of type qq` and a question of type q cannot. The dash at the end of (3.1) was placed where Chomsky would have put a so-called trace.

Traces are not needed in our approach and are only indicated to facilitate comparison with Chomsky's approach. It will turn out that a double superscript I makes an appearance whenever Chomsky requires a trace.

how a sentence such as (3.1) is analyzed in real time

(3.1)

$$\begin{array}{l} \textit{whom will she see} - ? \\ (\overline{\mathbf{q}}\mathbf{o}^{\ell\ell}\mathbf{q}^{\ell})(\mathbf{q}_1\mathbf{j}^{\ell}\pi^{\ell})\pi_3(\mathbf{i}\mathbf{o}^{\ell}) \quad \rightarrow \quad \overline{\mathbf{q}} \end{array}$$

$$\begin{aligned} \textit{whom}: &\quad \overline{\mathbf{q}}\hat{\mathbf{o}}^{\ell\ell}\mathbf{q}^{\ell}, \\ \textit{whom will}: &\quad \overline{\mathbf{q}}\hat{\mathbf{o}}^{\ell\ell}\,\underline{\mathbf{q}^{\ell}\mathbf{q}_1}\,\mathbf{j}^{\ell}\pi^{\ell} \;\rightarrow\; \overline{\mathbf{q}}\hat{\mathbf{o}}^{\ell\ell}\mathbf{j}^{\ell}\pi^{\ell}, \\ \textit{whom will she}: &\quad \overline{\mathbf{q}}\hat{\mathbf{o}}^{\ell\ell}\mathbf{j}^{\ell}\pi^{\ell}\pi_3 \;\rightarrow\; \overline{\mathbf{q}}\hat{\mathbf{o}}^{\ell\ell}\mathbf{j}^{\ell}, \\ \textit{whom will she see}: &\quad \overline{\mathbf{q}}\hat{\mathbf{o}}^{\ell\ell}\,\underline{\mathbf{j}^{\ell}\,\overbrace{\mathbf{i}\,\mathbf{o}^{\ell}}} \;\rightarrow\; \overline{\mathbf{q}}\,\underline{\hat{\mathbf{o}}^{\ell\ell}\mathbf{o}^{\ell}} \;\rightarrow\; \overline{\mathbf{q}}. \end{aligned}$$

Part IV

意味の形式的理論



小学生の算数を考える

小学算数の筆算のルールを考える

思考実験1

二桁の数どうしの計算について、小学校で習った足し算・引き算・掛け算の計算の仕方を、すべて「文章」で記述してみよう。
それぞれ、400字以内にまとめられるか？

加減乗除の筆算の例

$$\begin{array}{r} 64 \\ +\ 37 \\ \hline 101 \end{array}$$

$$\begin{array}{r} 64 \\ \times\ 37 \\ \hline 448 \\ 192\ \ \\ \hline 1368 \end{array}$$

$$\begin{array}{r} 64 \\ -\ 37 \\ \hline 27 \end{array}$$

$$\begin{array}{r} \ \ 2 \\ \hline 37)\ 64 \\ \ \ 64 \\ \hline \ \ 0 \end{array}$$

そのルールを書き出してみよう

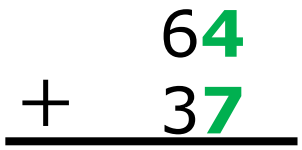
足し算の筆算のルール

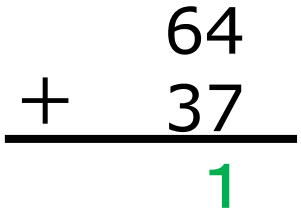
1. 一番下の桁の二つの数を足す。
   4+7 = 11

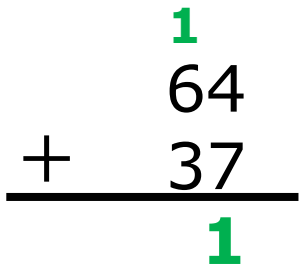
2. その答えの一番下の桁の数字を下に書く。
   4+7 = 1**1**

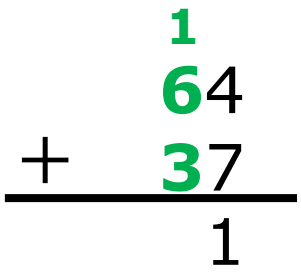
3. その答えが、10未満だったら、次の桁に進む。
   10以上だったら、次の桁の上に、小さく、繰り上がりの1を書く。

4. 次の桁の二つの数と、繰り上がりがあれば、それも合わせてたす。
   1+6+3=10









5. その答えの一番下の桁の数字を下に書く。 1+6+3 = 1**0**

$$\begin{array}{r} {\scriptstyle 1} \\ 64 \\ +\ 37 \\ \hline \mathbf{0}1 \end{array}$$

6. その答えが、10未満だったら、次の桁に進む。
10以上だったら、次の桁の上に、小さく、繰り上がりの1を書く。

$$\begin{array}{r} {\scriptstyle \mathbf{1}\ 1} \\ 64 \\ +\ 37 \\ \hline 01 \end{array}$$

7. 次の桁の二つの数と、繰り上がりがあれば、それも合わせてたす。
1+0+0=1

$$\begin{array}{r} {\scriptstyle \mathbf{1}\ 1} \\ \mathbf{0}64 \\ +\ \mathbf{0}37 \\ \hline 01 \end{array}$$

8. その答えの一番下の桁の数字を下に書く。 1+0+0 = **1**

$$\begin{array}{r} {\scriptstyle 1\ 1} \\ 64 \\ +\ 37 \\ \hline \mathbf{1}01 \end{array}$$

9. その答えが、10未満だったら、
次の桁に進む。
10以上だったら、次の桁の上に、
小さく、繰り上がりの1を書く。

```
  1 1
   64
+  37
-----
  101
```

10.たすものがなければ、計算終了。

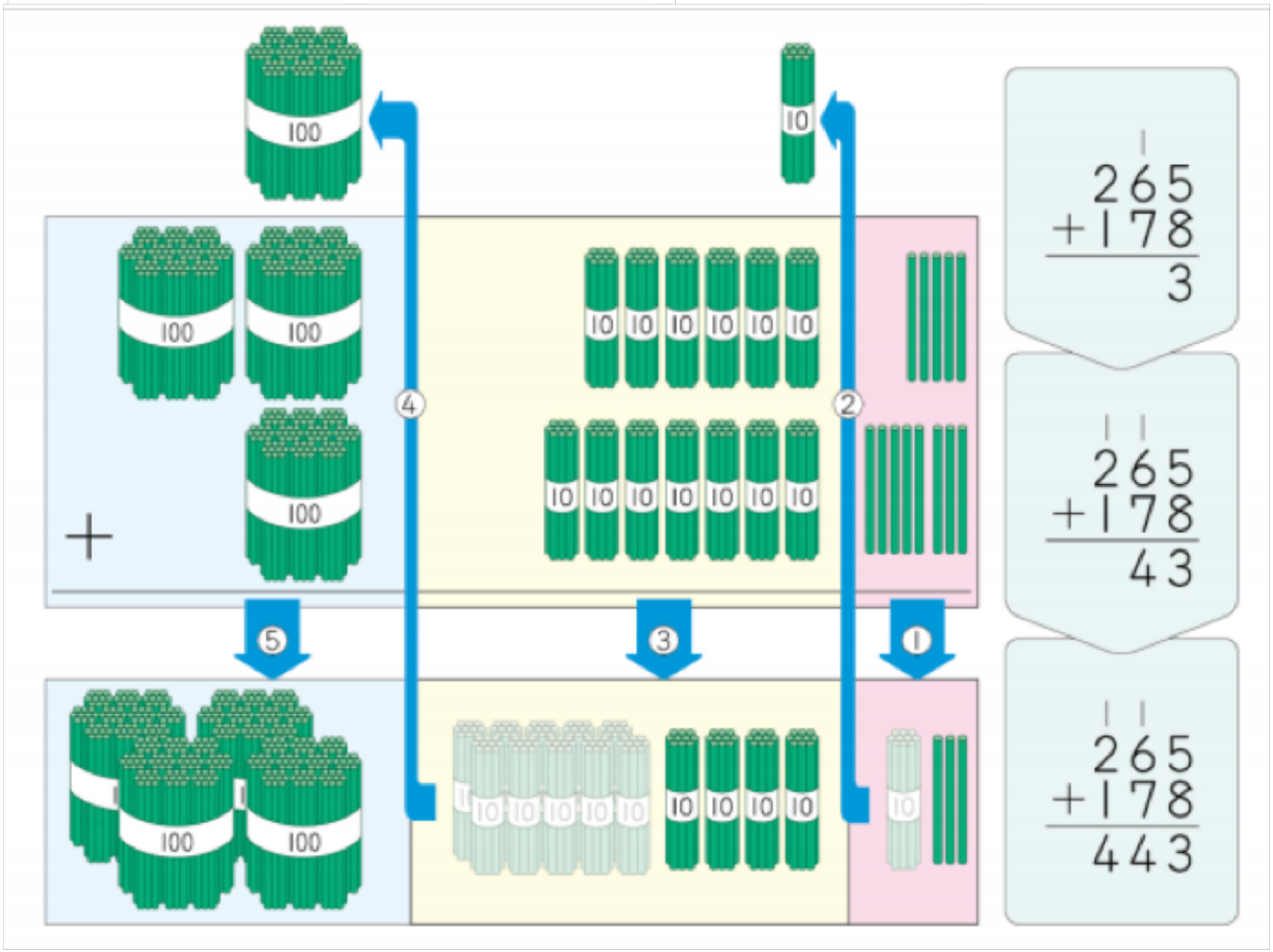
```
   64
+  37
-----
  101
```

足した結果の、一番下の桁の数字を書き込む
足した結果が、10以上か（繰り上がり）、
そうでないか（繰り上がりなし）で処理が違う。

「桁」って、何？
足し算の九九が前提

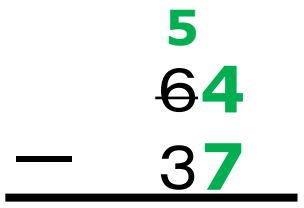
3 265＋178 の筆算のしかたを考えましょう。

啓林館　3年上 p.57

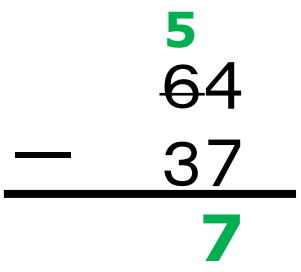


引き算の筆算のルール

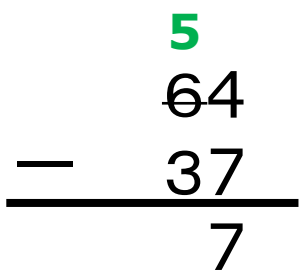
1. 一番下の桁の二つの数を引く。
   引けない場合は、一つ上の桁を
   一つ減らして、次の計算を行う。
   14-7 = 7
   4+(10-7) = 7



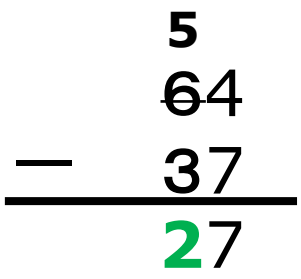
2. その答えの数字を下に書く。



3. 次の桁の二つの数を引く。
   5-3=2
   引けない場合は、一つ上の桁を
   一つ減らして計算する。



4. その答えの数字を下に書く。



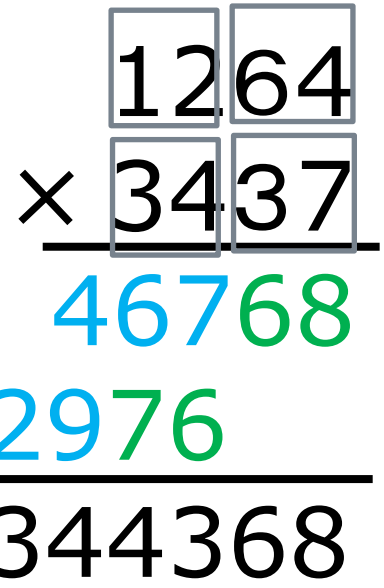
掛け算の筆算のルール

1. 7 x 4 を計算する。
   7 x 4 = 28
2. この答え28の一番下の数字 **8** を
   書き込み、上の数字2は覚えておく。
3. 7 x 6 を計算する。
   7 x 6 = 42
4. この42に、先に覚えた2 を足して
   42 + 2 = **44** を、上の桁に書き込む。
5. 3 x 4 を計算する、
   3 x 4 = 12
6. この答えの一番下の数字 **2** を、
   3の下に書き込み、上の数字1は覚えておく。
7. 3 x 6 を計算する。
   3 x 6 = 18
8. この18に、先に覚えた1を足して、18+1 =**19**を、
   上の桁に書き込む。
9. **448**と**1920**を足し算する。……

$$\begin{array}{r} 64 \\ \times\ 37 \\ \hline 448 \\ 192 \\ \hline 2368 \end{array}$$

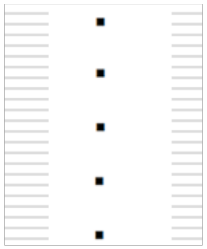
もしも、二桁の数字の
かけ算が暗算でできたら

1. 37 x 64 を計算する。
   37 x 64 = 2368
2. この答え2368の下の二桁の数字 **68** を
   書き込み、上の数字23は覚えておく。
3. 37 x 12 を計算する。
   37 x 12 = 444
4. この444に、先に覚えた23 を足して
   444 + 23 = **467** を、上の桁に書き込む。
5. 34 x 64 を計算する、
   34 x 64 = 2176
6. この答えの下の二桁の数字 **76** を、
   3桁の下に書き込み、上の数字21は覚えておく。
7. 34 x 12 を計算する。
   34 x 12 = 408
8. この408に、先に覚えた21を足して、408+21 =**429**を、
   上の桁に書き込む。
9. **46768**と**4297600**を足し算する。……



足し算でかけ算をする(ちょっと馬鹿げている)

1. 64+0=64
2. 64+64=128
3. 64+128=192
4. 64+192=256



35. 64+2176=2240
36. 64+2240=2304
37. 64+2304=2368

割り算の筆算のルール？

```
  1176
2)2352
  2
  3
  2
  15
  14
   12
   12
    0
```

```
  112
21)2352
  21
   25
   21
   42
   42
    0
```

```
   42
56)2352
  224
  112
  112
    0
```

```
    8
294)2352
 2352
    0
```

かけ算で行う割り算のアルゴリズム(Silly!)

2x1=2
2x2=4
2x3=6
2x4=8
.
.
.
.
.
.
.
.
.
2x1173=2346
2x1174=2348
2x1175=2350
2x1176=2352
2x1177=2354
.
.
.

21x1=21
21x2=42
21x3=63
21x4=84
.
.
.
.
.
.
.
21x109=2289
21x110=2310
21x111=2331
21x112=2352
21x113=2373
.
.
.

56x1=56
56x2=112
56x3=168
56x4=184
.
.
.
56x39=2184
56x40=2240
56x41=2396
56x42=2352
56x43=2408
.
.
.

294x1=294
294x2=588
294x3=882
294x4=1176
294x5=1470
294x6=1764
294x7=2058
294x8=2352
294x9=2646
.
.
.

計算をたとえで説明する

思考実験

「二分の一を三分の二で割る」を、「例え話」で説明せよ。

次の分数の計算を「たとえ」で説明せよ

- $\frac{1}{2} + \frac{2}{3} = \frac{7}{6}$
- $\frac{2}{3} - \frac{1}{2} = \frac{1}{6}$
- $\frac{1}{2} \times \frac{2}{3} = \frac{1}{3}$
- $\frac{1}{2} \div \frac{2}{3} = \frac{3}{4}$

分数計算の「ルール」

- $\frac{1}{2} + \frac{2}{3} = \frac{7}{6}$
- $\frac{2}{3} - \frac{1}{2} = \frac{1}{6}$

二つの分母を「通分」して、分母を同じにしてから、分子の足す・引くを行う。

- $\frac{1}{2} \times \frac{2}{3} = \frac{1}{3}$

分子は分子、分母は分母でかける。

- $\frac{1}{2} \div \frac{2}{3} = \frac{3}{4}$

割る数の分子と分母を逆にして割られる数にかける。

$\frac{1}{2} + \frac{2}{3} = \frac{7}{6}$ の場合

12本の鉛筆が一つのケースに入っています。
ケースを二等分すれば、1/2ケースには、鉛筆6本が入っています。ケースを三等分すれば、1/3ケースには、鉛筆4本が入っています。2/3ケースは、1/3ケースの二倍ですので、2/3ケースには、鉛筆が8本入ります。

ですので、(1/2+2/3)ケースには、鉛筆が、6+8=14本入ることになります。

これは、ケース一つ（12本）とケース6分の1(2本)分になります。

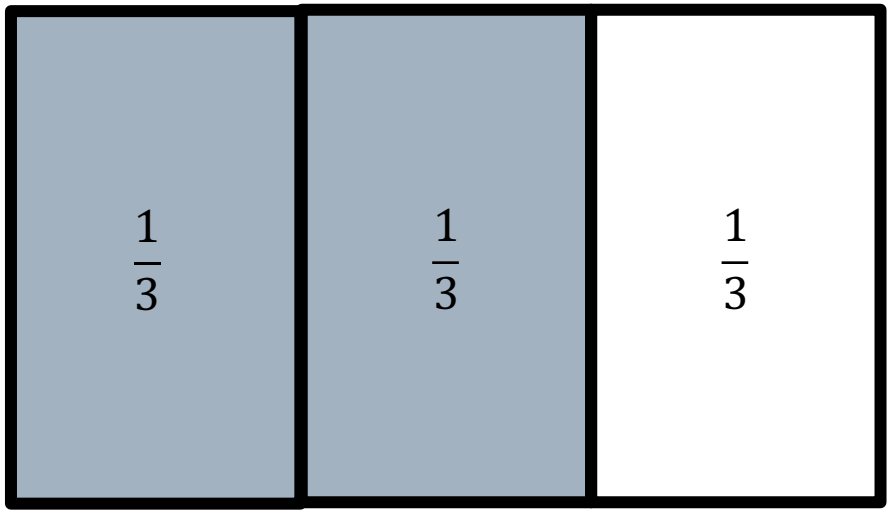
$\frac{2}{3} - \frac{1}{2} = \frac{1}{6}$ の場合

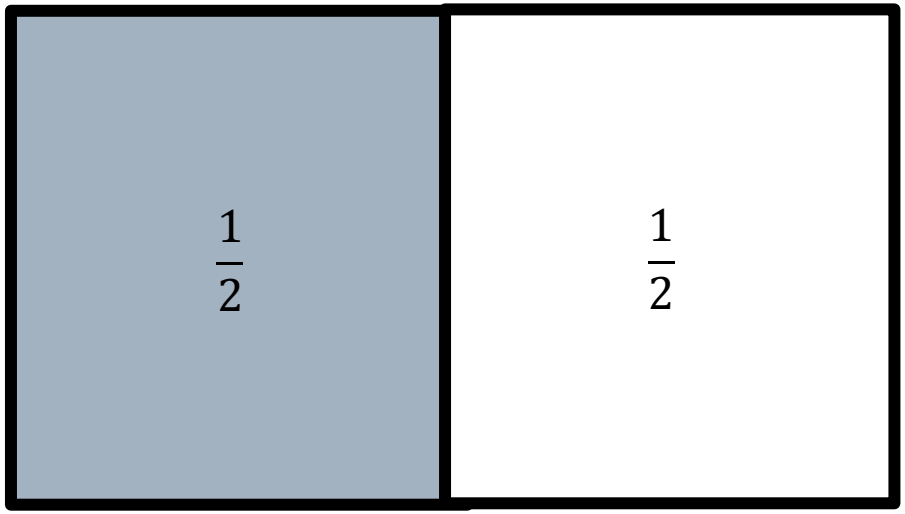
12本の鉛筆が一つのケースに入っています。
ケースを二等分すれば、1/2ケースには、鉛筆6本が入っています。ケースを三等分すれば、1/3ケースには、鉛筆4本が入っています。2/3ケースは、1/3ケースの二倍ですので、2/3ケースには、鉛筆が8本入ります。

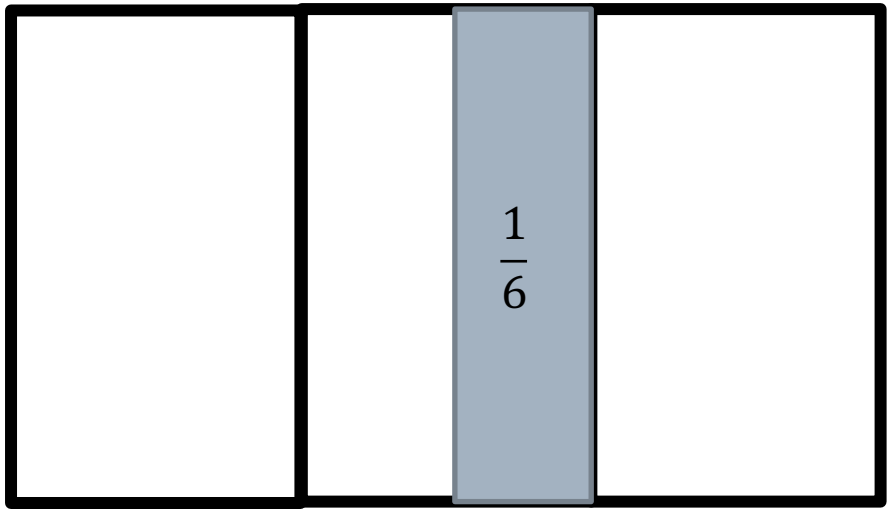
ですので、(2/3−1/2)ケースには、鉛筆が、8-6=2本入ることになります。

これは、ケース一つ（12本）の1/6になります。

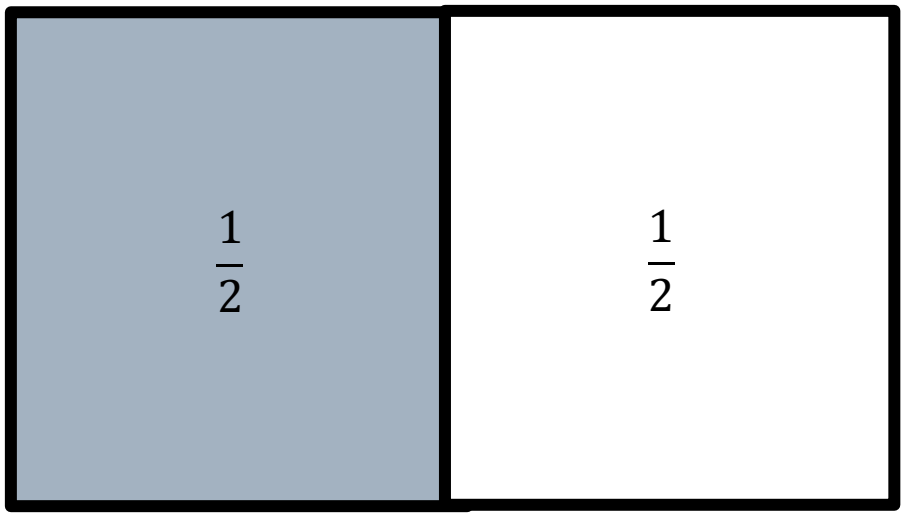
$\frac{2}{3} - \frac{1}{2} = \frac{1}{6}$ の場合

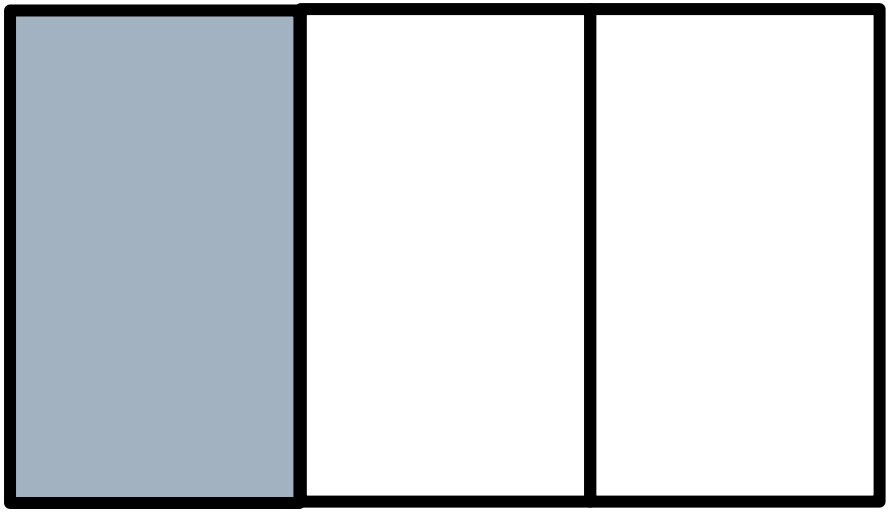


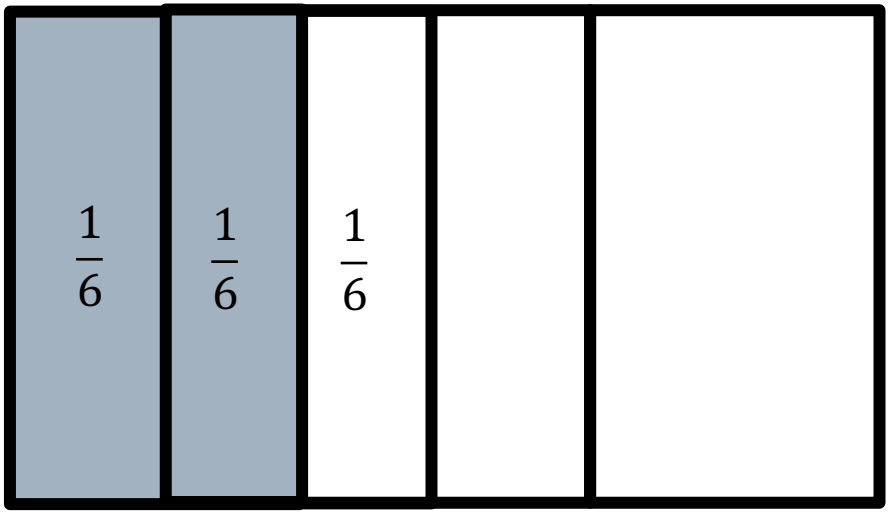




$\frac{1}{2} \times \frac{2}{3} = \frac{2}{6} = \frac{1}{3}$ の場合







$\frac{1}{2} \div \frac{2}{3} = \frac{3}{4}$ の場合

鉛筆が12本入ったケースと、3人で1組のグループがあったとする。この時、ケースの1/2をグループの2/3で分けたとしよう。

これは、鉛筆6本を二人で分けたことになるので、グループの一人は3本(3/12)の鉛筆を取ったことになる。この取り方で、一つのグループあたり何本の鉛筆が分配されたかを考えれば、グループは、3人で1組なので、9本の鉛筆をえたことになる。これは、ケースで考えれば、9/12=3/4ケースになる。

1グループあたりのケースを求めると、

$$\frac{1}{2} \div \frac{2}{3} = \frac{1}{2\times2} \times 3 = \frac{1\times3}{2\times2} = \frac{3}{4}$$

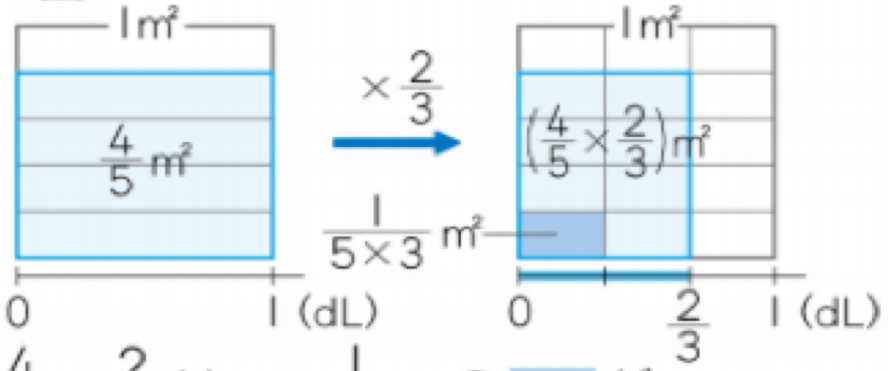
6年 p.280

4 1dLで$\frac{4}{5}$m²ぬれるペンキがあります。
$\frac{2}{3}$dLのペンキでは何m²ぬれますか。





ひなた



$\frac{4}{5}\times\frac{2}{3}$ は，$\frac{1}{5\times3}$ の［　］が

(4×2)個分だから，

$$\frac{4}{5}\times\frac{2}{3}=\frac{\square\times\square}{\square\times\square}$$

$$=\square$$

［　］m²



だいち

$$\frac{4}{5}\times\frac{2}{3}=\frac{4}{5}\times\left(\frac{1}{3}\times2\right)$$

$$=\left(\frac{4}{5}\times\frac{1}{3}\right)\times2$$

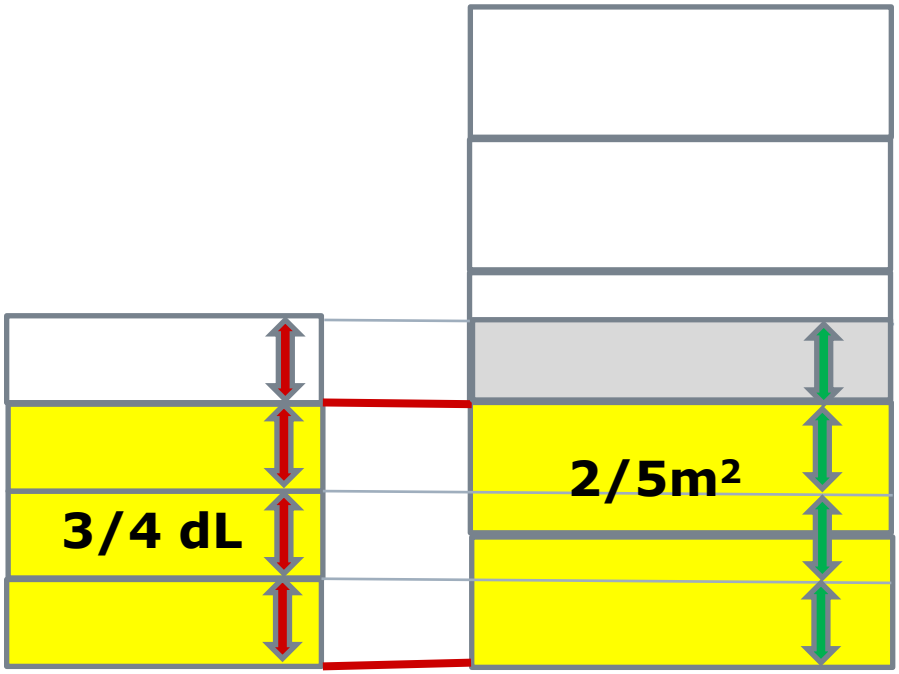
$$=\frac{4}{5\times3}\times2$$

$$=\frac{\square\times\square}{\square\times\square}$$

$$=\square$$

［　］m²

3/4dLのペンキで2/5m² ぬるとき，1dLでぬれる面積を求める式を考える

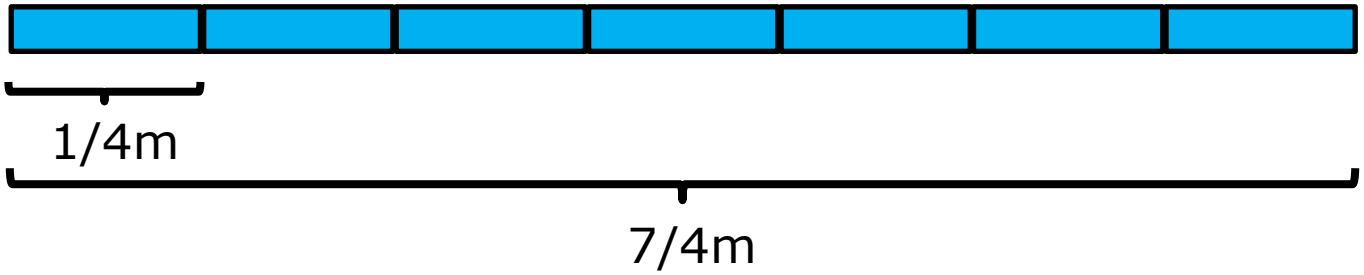


もしも、2dLのペンキで、3m² ぬれるのなら、1dLでぬれる面積は、3　2の1.5m² である。
だから、この場合の式は、
$\frac{2}{5}$　$\frac{3}{4}$となる。

1/4dLでぬれる面積は、2/5m²の1/3なので、2/5x1/3=2/15m²
よって3/4dLでぬれる面積は、
2/15x4=8/15

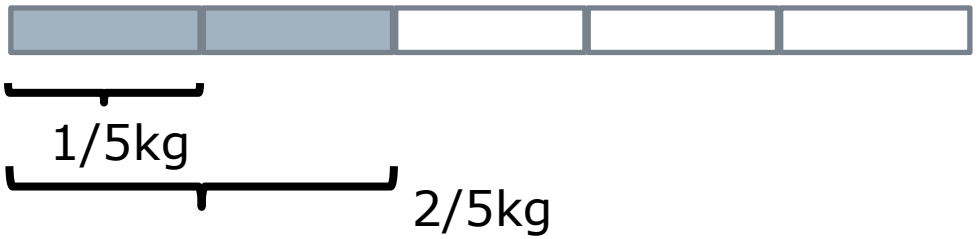
$$\frac{2}{5}\quad\frac{3}{4}=\frac{2}{5}\times\frac{1}{3}\times 4$$
$$=\frac{2}{5}\times\frac{4}{3}$$
$$=\frac{8}{15}$$

7/4mの重さが2/5kgのホースについて、
ホース1mの重さ，及びホース1kgの長さを求めよ



7/4mの重さが2/5kg だから、1/4mの重さは、2/5　7=2/35kg
よって、1m=4/4mの重さは、2/35x4=8/35kg。

$$\frac{2}{5} \div \frac{7}{4} = \frac{2}{5} \times \frac{4}{7} = \frac{8}{35}$$



2/5kgの長さが7/4m だから、1/5kgの長さは、7/4　2=7/8
よって、1kg=5/5kgの長さは、7/8x5=35/8m。

$$\frac{7}{4} \div \frac{2}{5} = \frac{7}{4} \times \frac{5}{2} = \frac{35}{8}$$

「ルール」の世界と「たとえ」の世界

「ルール」で理解する vs.「たとえ」で理解する

2+3が5になるのを、「りんご二つと、みかん三つで、全部で五つ」というように、具体的なものをあげて、「たとえ」で説明することがある。

数学のルールは、基本的には現実の具体的な関係から抽象されたものであるので、理解しやすい具体的関係から、ルールを導入・説明するのは有効である。

「ルール」は、抽象的なものだが、「たとえ」は、ルールの「意味」を理解させる。

「ルール」で理解する vs.「たとえ」で理解する

先のりんごとみかんの例では、実は、「全部で」という言葉が、足し算の説明になっている。「たとえ」では、具体的なものにたいする理解とともに、「言葉」の理解が、前提になることがある。

「たとえ」の世界は、具体的で現実的である。その世界は、我々の直観と「言葉」によって理解可能な「意味」の世界によって支えられている。

ただ、それは、複数の意味を持つかもしれないし、ひとつの「たとえ」で全ての「ルール」を説明できないという意味では個別的である。

また「たとえ」にはいろいろのタイプがある。言葉だけでなく、図解することも、「たとえ」による説明の一種である。言葉による「たとえ」と図形による「たとえ」では、抽象度が異なる。

「ルール」の世界と「たとえ」の世界

数学の計算では「ルール」を「アルゴリズム」という。その「ルール」に従って、きちんと計算すると、余計なことは考えなくても正しい計算結果が保証されるという限りでは、「ルール」の世界は、他の世界とは独立しているように見える。

ただ、「ルール」は、計算をするという目的のためには万能かもしれない。ただ、計算「ルール」というものは、具体的なようで我々の頭の中にしかない抽象的なものであって、それ自体では意味を持たないようにも見える。

数学は、基本的には、それ自身の原理に基づいた自律的な「ルール」の世界である。（実際には、それは単純化しすぎである。）

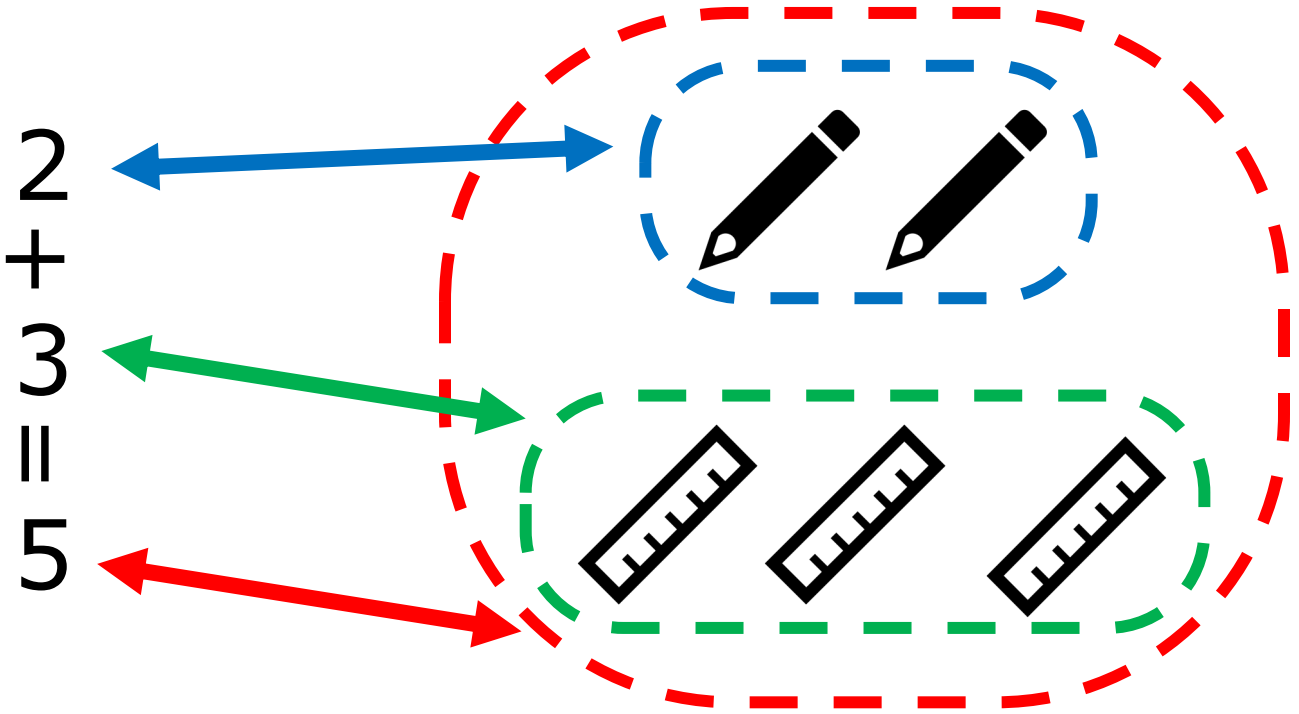
理論とモデル

「ルール」と「たとえ」の数理

素朴な対応理論

「ルール」と「たとえ」の対応関係

先に見た「ルールの世界」と「たとえの世界」の二つの世界は、お互いに関連している。「ルールの世界」は「たとえの世界」に、具体的対応物を持ち、「たとえの世界」は「ルールの世界」に抽象的な対応物を持つ。

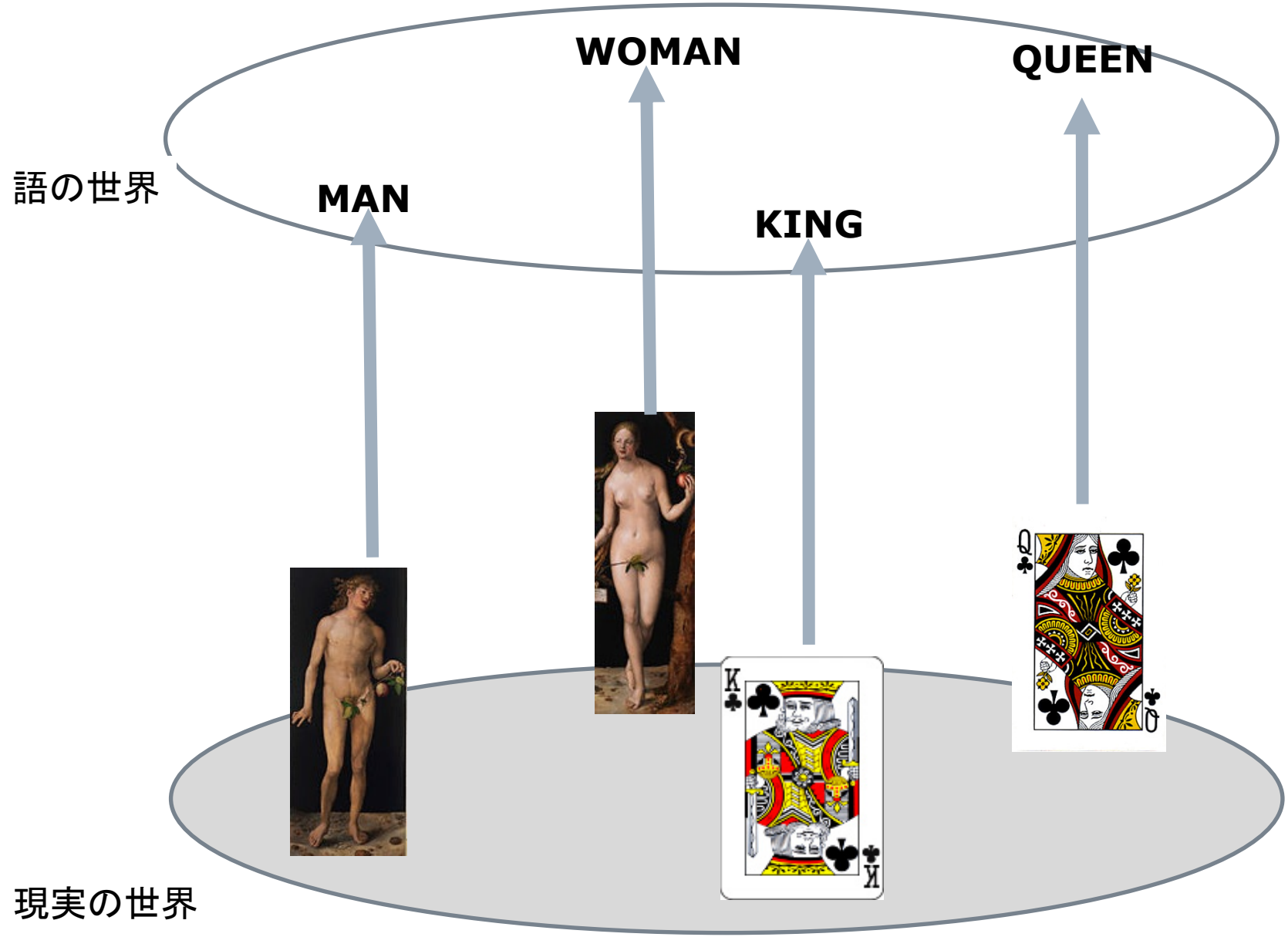


二つの「世界」の対応関係が、「意味」を与える

二つの世界に対応関係があり、一方の世界がより具体的で、もう一方の世界がより抽象的な場合、この対応関係は、抽象的な世界に、具体的な「意味」を与える。あるいは、この対応関係は、抽象的なものに具体的な「解釈」を与える。

以下では、「語の意味」の素朴なモデルとして、現実に存在する「事物」と、その「名前」の「対応関係」を、考えてみよう。

「語の意味」の素朴なモデル



意味：ものに「名前」をつけること

語の世界

WOMAN

QUEEN

MAN

KING

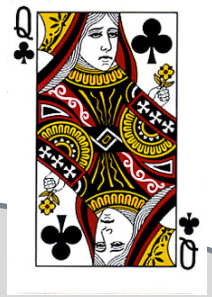
対応づける
名付ける

対応づける
名付ける









現実の世界

語は、ものを「参照」する

語の世界

WOMAN

QUEEN

MAN

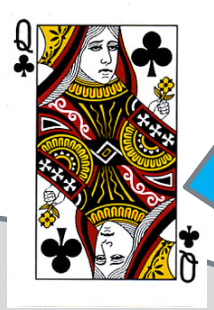
KING

参照する

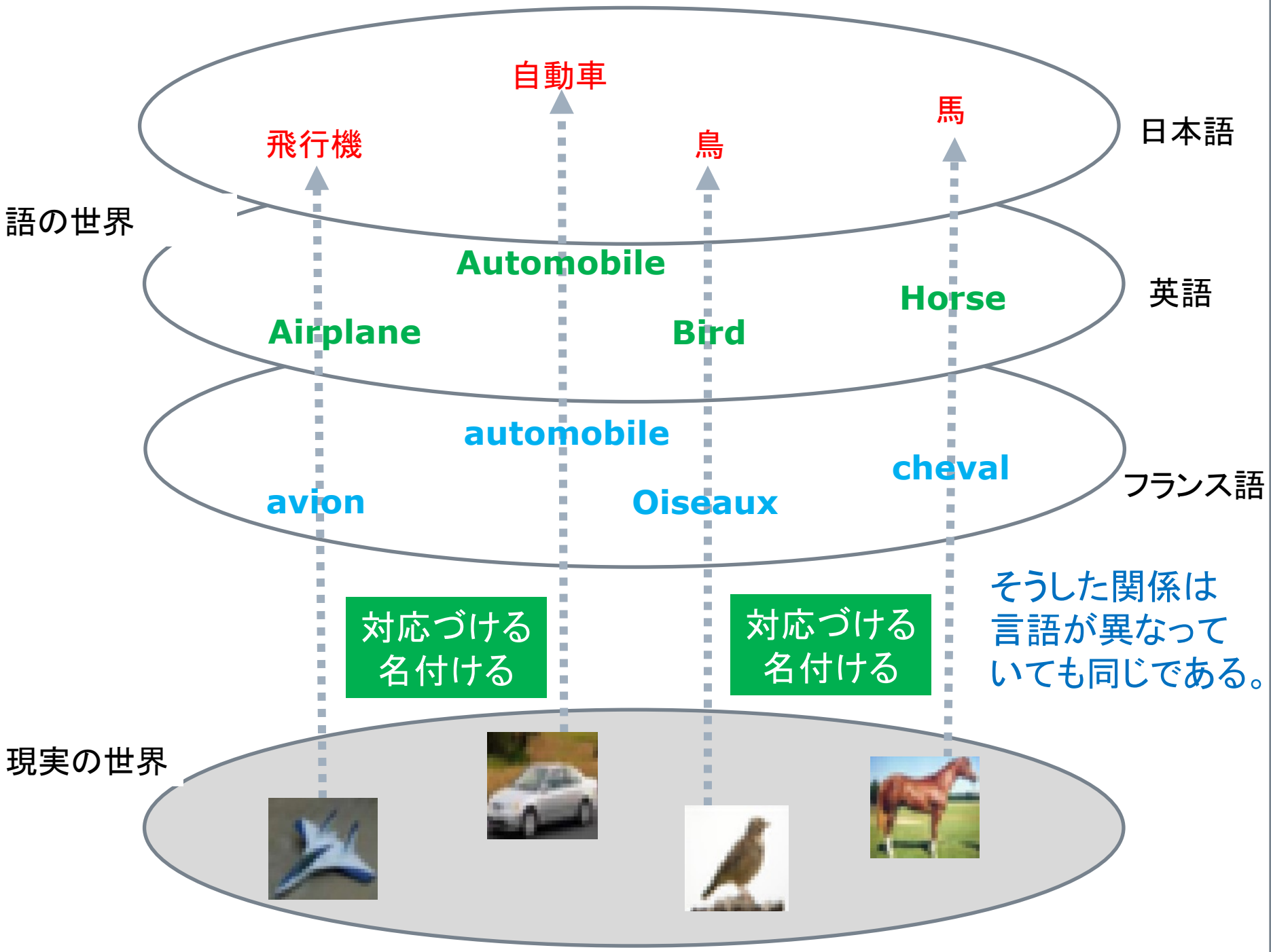


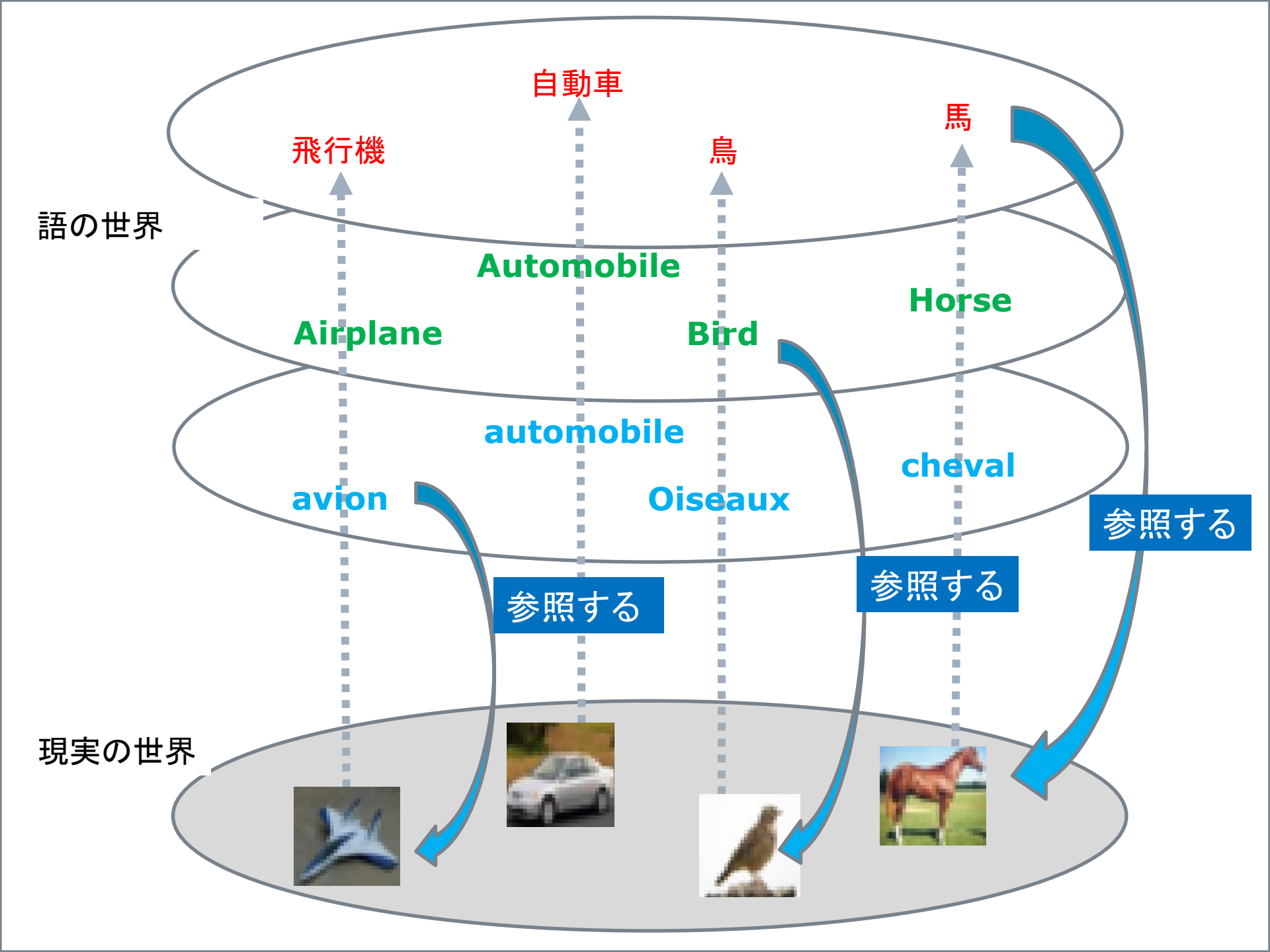






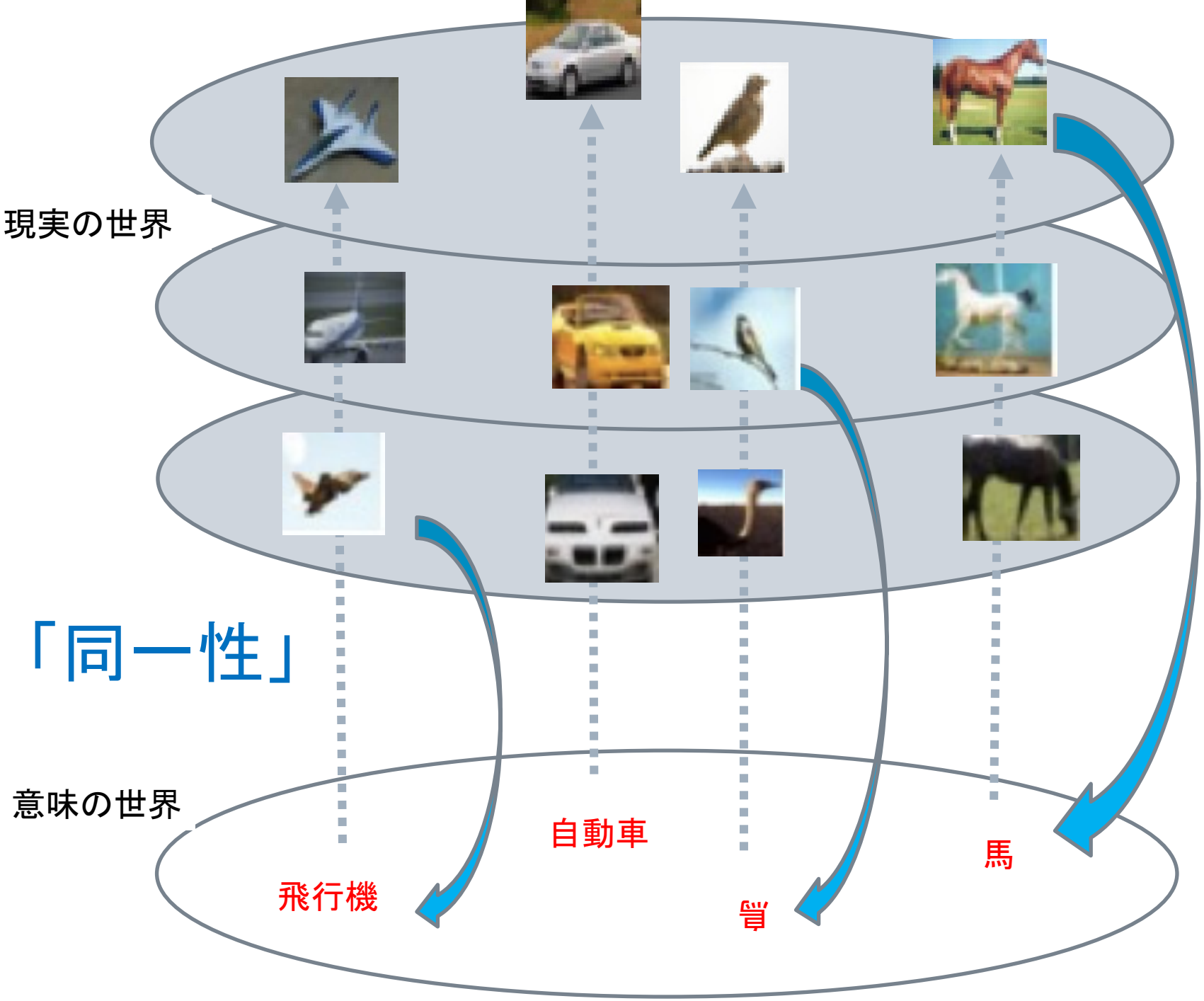
現実の世界


自動車
飛行機
鳥
馬
日本語
語の世界
Automobile
Horse
英語
Airplane
Bird
automobile
cheval
フランス語
avion
Oiseaux
対応づける
名付ける
対応づける
名付ける
そうした関係は
言語が異なって
いても同じである。
現実の世界


語の世界
自動車
飛行機
鳥
馬
Automobile
Airplane
Bird
Horse
automobile
avion
Oiseaux
cheval
参照する
参照する
参照する
現実の世界

素朴な「対応理論」の弱点

- ただ、こうした素朴な「対応理論」には、弱点もある。「現実の世界での『馬』」という対象を考えてみよう。こうした対象の「実在性」については、は、いろいろな疑問がうまれる。「現実に存在するのは、個々の具体的なウマだけである。」「いや、『馬』の意味の実在性には、疑問の余地はない。」等々。
- その上、語は（文になればなおさらなのだが）、「物」に付けられた意味を持つ名前だけではない。抽象的な概念（「自由」「平等」「意味」…）や、物ではない「出来事」(「第二次世界大戦」「爆発」…)も名前を持つのだ。具体的な心理的な状態（「悲しみ」「意図」…）も。これらを先の図にマッピングするのは、難しい。
- 「状態の変化」の一群は、「動詞」で表現される。


現実の世界
「同一性」
意味の世界
飛行機
自動車
鳥
馬

理論とモデル

対応の形式化

「理論」と「モデル」の形式的定義

ここでは、「ルールの世界」と「たとえの世界」の数学的形式化を紹介したい。

ある世界で、基本的な公理と一階の述語論理の演繹規則が与えられているとしよう。この世界を「理論 Theory」と呼ぶことにしよう。この「理論」の世界で、ある命題φが成り立つのは、この理論の中で、命題φが証明される場合に限る。

今度は、もう一つ別の世界を考えよう。この世界では、ある命題φが妥当する（「真」である）ことが、何らかの方法で定義されているとする。こうした世界を「モデル Model」と呼ぼう。

ある理論TとあるモデルMについて、すべての命題φについて、次の関係が成り立つとする。

Tにおいてφは証明可能 → Mにおいてφは真である

この時、TはモデルMを持つ。あるいは、MはTのモデルである。という。

ゲーデルの完全性定理

理論とモデルの関係を、「すべてのモデルで真になる命題は、一階の述語論理で証明可能である」という形で初めて述べたのは、ゲーデルである。これをゲーデルの「完全性定理」という。

この定理から、「ある理論が無矛盾ならそれはモデルを持つ」ことが導かれる。https://goo.gl/5HLMsK

モデルの存在定理では、次の「レーベンハイム＝スコーレムの定理」が有名である。

「理論が非可算濃度のモデルを持つなら、それは可算モデルも持つ。」

Appendix



Lambek 1958
Lambek 2008
IBM Watson

The mathematics of sentence structure

JOACHIM LAMBEK

http://lecomte.al.free.fr/ressources/PARIS8_LSL/Lambek.pdf

Lambek (1958)論文の数学的側面

- Lambek (1958)論文は、言語学の論文であるが、数学的側面を持つ。
- その一つは、先に見たいくつかの計算ルールが（青い枠で囲った部分）が、より基本的なルールから論理的に導出できることを明らかにしたことである。その、より基本的なルールの体系を、彼は”Syntactic Calculus”と呼ぶ。
- もう一つは、このSyntactic Calculusは、「決定可能」な体系であることを、数学的に証明したことである。ここでは、この部分は、割愛した。

7. Syntactic calculus

文法の最も基本的なルール。これから多くの計算ルールが導かれる。

(a) $x \to x$

(b) $(xy)z \to x(yz)$ (b') $x(yz) \to (xy)z$

(c) if $xy \to z$ then $x \to z/y$ (c') if $xy \to z$ then $y \to x\backslash z$

(d) if $x \to z/y$ then $xy \to z$ (d') if $y \to x\backslash z$ then $xy \to z$

(e) if $x \to y$ and $y \to z$ then $x \to z$

先のルールから導かれるべき、
いくつかの計算ルール。

(f) $x \to (xy)/y$,

(g) $(z/y)y \to z$,

(h) $y \to (z/y)\backslash z$,

(i) $(z/y)(y/x) \to z/x$,

(j) $z/y \to (z/x)/(y/x)$,

(k) $(x\backslash y)/z \longleftrightarrow x\backslash(y/z)$,

(l) $(x/y)/z \longleftrightarrow x/(zy)$,

(m) if $x \to x'$ and $y \to y'$ then $xy \to x'y'$,

(n) if $x \to x'$ and $y \to y'$ then $x/y' \to x'/y$.

Proof of (m).

$$\dfrac{\dfrac{x \to x' \quad \dfrac{x'y \to x'y}{x' \to (x'y)/y}\,(\mathrm{c})}{x \to (x'y)/y}\,(\mathrm{e})}{xy \to x'y}\,(\mathrm{d}) \qquad \dfrac{\dfrac{y \to y' \quad \dfrac{x'y' \to x'y'}{y' \to x'\backslash(x'y')}\,(\mathrm{c'})}{y \to x'\backslash(x'y')}\,(\mathrm{e})}{x'y \to x'y'}\,(\mathrm{d'})$$

$$\dfrac{xy \to x'y \qquad x'y \to x'y'}{xy \to x'y'}\,(\mathrm{e})$$

Proof of (n).

$$\dfrac{\dfrac{y \to y' \quad \dfrac{\dfrac{\dfrac{x'y \to x'y}{(x/y')y' \to x}\,(\mathrm{d})}{y' \to (x/y')\backslash x}\,(\mathrm{c'})}{y \to (x/y')\backslash x}\,(\mathrm{e})}{(x/y')y \to x}\,(\mathrm{d'})}{x/y' \to x/y}\,(\mathrm{c}) \qquad \dfrac{\dfrac{\dfrac{x/y \to x/y}{(x/y)y \to x}\,(\mathrm{d}) \qquad x \to x'}{(x/y)y \to x'}\,(\mathrm{e})}{x/y \to x'/y}\,(\mathrm{d})}{x/y' \to x'/y}\,(\mathrm{e})$$

From word to sentence: a computational algebraic approach to grammar

Joachim Lambek

http://www.math.mcgill.ca/barr/lambek/pdffiles/2008lambek.pdf

I. Introduction

1. Origin of languages.
2. Grammar and mathematics.
3. An algebraic approach to grammar.
4. Mathematical interlude.

II. Basic Word Classes

5. Noun.
6. Philosophical digression.
7. Adjectives.
8. Verbs.
9. Adverbs.

5. Nouns.

In addition to names, such as *John* and *Napoleon*, there are three kinds of nouns in English:

count nouns: *bean, pig, idea, delivery*, ...

mass nouns: *rice, pork, water, sadness*, ...

plurals: *beans, pigs*, ..., *police, cattle*, ..., *pants, scissors*, ...

n = name,

$\mathbf{n}_0$ = mass noun,

$\mathbf{n}_1$ = count noun,

$\mathbf{n}_2$ = plural.

Napoleon slept,

$\underline{\mathbf{n}\ (\pi_3^r}\ \mathbf{s}_2) \to \mathbf{s}_2$

I see John

$\underline{\pi_1\ (\pi_1^r}\ \mathbf{s}_1\ \underline{\mathbf{o}^\ell)\mathbf{n}} \to \mathbf{s}_1$

$$\mathbf{n} \to \pi_3,\ \mathbf{o}$$

$$\mathbf{n}\pi_3^r \to \pi_3\pi_3^r \to 1.$$

$\mathbf{s}_2 =$ declarative sentence in past tense,
$\pi_1 =$ first person subject,
$\pi_2 =$ plural or second person subject.

$$\pi_k \to \pi \quad (k = 1, 2, 3).$$

$$\underline{\mathbf{n}_0\ (\pi_3^r}\ \mathbf{s}_1\ \underline{\overline{\mathbf{a}}^\ell)\mathbf{a}} \quad \to \quad \mathbf{s}_1 \qquad \underline{\pi_1\ (\pi^r}\ \mathbf{s}_2\ \underline{\mathbf{o}^\ell)(\overline{\mathbf{n}}_0}\ \underline{\mathbf{n}_0^\ell)\ \mathbf{n}_0} \quad \to \quad \mathbf{s}_2$$

water is wet,

I drank much water.

$$\begin{aligned}
\mathbf{a} &= \text{ predicative adjective,} \\
\overline{\mathbf{a}} &= \text{ predicative adjectival phrase,} \\
\overline{\mathbf{n}}_i &= \text{ complete noun phrase.}
\end{aligned}$$

$$\begin{aligned}
&\mathbf{a} \to \overline{\mathbf{a}} \not\to \mathbf{a}, \\
&\mathbf{n}_0 \to \overline{\mathbf{n}}_0 \to \pi_3,\ \mathbf{o}, \\
&\mathbf{n}_2 \to \overline{\mathbf{n}}_2 \to \pi_2,\ \mathbf{o}, \\
&\overline{\mathbf{n}}_1 \to \pi_3,\ \mathbf{o}.
\end{aligned}$$

$$\textit{the king is dead},$$
$$(\overline{\mathbf{n}}_1\,\underbrace{\mathbf{n}_1^{\ell})\;\mathbf{n}_1}\;\underbrace{(\pi_3^{r}\,\mathbf{s}_1\,\overline{\mathbf{a}}^{\ell})\mathbf{a}} \;\rightarrow\; \mathbf{s}_1$$

$$\textit{she met a boy}.$$
$$\underbrace{\pi_3\;(\pi^{r}}\,\mathbf{s}_2\,\underbrace{\mathbf{o}^{\ell})(\overline{\mathbf{n}}_1}\,\underbrace{\mathbf{n}_1^{\ell})\mathbf{n}_1} \;\rightarrow\; \mathbf{s}_2$$

$$\textit{kings will die},$$
$$\underbrace{\mathbf{n}_2\;(\pi^{r}}\,\mathbf{s}_1\,\underbrace{\mathbf{j}^{\ell})\mathbf{i}} \;\rightarrow\; \mathbf{s}_1$$

$$\mathbf{n}_2 \rightarrow \overline{\mathbf{n}}_2 \rightarrow \pi_2 \rightarrow \pi,$$

$$\textit{they have many pigs}\ .$$
$$\underbrace{\pi_2\;(\pi_2^{r}}\,\mathbf{s}_1\,\underbrace{\mathbf{o}^{\ell})(\overline{\mathbf{n}}_2}\,\underbrace{\mathbf{n}_2^{\ell})\;\mathbf{n}_2} \;\rightarrow\; \mathbf{s}_1$$

$$
\begin{array}{c}
a\ man\ eating\ fish, \\
(\overline{\mathbf{n}}_1\ \mathbf{n}_1^{\ell})\ [\underbrace{\mathbf{n}_1\ (\mathbf{n}_1^{r}}\ \mathbf{n}_1\ \underbrace{\mathbf{o}^{\ell})\ \mathbf{n}_0}
\end{array}
\quad \rightarrow \quad \overline{\mathbf{n}}_1
$$

$$
\begin{array}{c}
a\ (man - eating)\ fish, \\
(\overline{\mathbf{n}}_1\ \underbrace{\mathbf{n}_1^{\ell})\ (\mathbf{n}_1}\ \underbrace{\mathbf{n}_1^{\ell})\ \mathbf{n}_1}
\end{array}
\quad \rightarrow \quad \overline{\mathbf{n}}_1
$$

7. Adjectives

- Adjectives in English play two distinct rˆoles. They can serve as predicates, as in

$$\begin{array}{l} \textit{she is poor}, \\ \underbrace{\pi_3\,(\pi_3^r\,\mathbf{s}_1\,\underbrace{\overline{\mathbf{a}}^{\ell}}\,)\mathbf{a}} \end{array} \quad \rightarrow \quad \mathbf{s}_1$$

- or as attributes modifying nouns, as in

$$\begin{array}{l} \textit{the poor man}, \\ (\overline{\mathbf{n}}_1\,\underbrace{\mathbf{n}_1^{\ell}})(\mathbf{n}_1\,\underbrace{\mathbf{n}_1^{\ell}})\mathbf{n}_1 \end{array} \quad \rightarrow \quad \overline{\mathbf{n}}_1 \qquad \begin{array}{l} \textit{poor men}, \\ (\mathbf{n}_2\,\underbrace{\mathbf{n}_2^{\ell}})\mathbf{n}_2 \end{array} \quad \rightarrow \quad \mathbf{n}_2, \qquad \begin{array}{l} \textit{poor wine}, \\ (\mathbf{n}_0\,\underbrace{\mathbf{n}_0^{\ell}})\mathbf{n}_0 \end{array} \quad \rightarrow \quad \mathbf{n}_0$$

the poor old man

$$(\overline{\mathbf{n}}_1\,\underline{\mathbf{n}_1^{\ell}})(\mathbf{n}_1\,\underline{\mathbf{n}_1^{\ell}})(\mathbf{n}_1\,\underline{\mathbf{n}_1^{\ell}})\mathbf{n}_1 \;\rightarrow\; \overline{\mathbf{n}}_1$$

**she is poor old,*

$$\underline{\pi_3(\pi_3^r}\,\mathbf{s}_1\,\underline{\overline{\mathbf{a}}^{\ell})\mathbf{a}}\;\mathbf{a} \;\rightarrow\; \mathbf{s}_1\mathbf{a} \not\rightarrow \mathbf{s}_1.$$

she is very poor

$$\underline{(\pi_3\,(\pi_3^r}\,\mathbf{s}_1\,\underline{\overline{\mathbf{a}}^{\ell})\,(\mathbf{a}}\,\underline{\mathbf{a}^{\ell})\mathbf{a}} \;\rightarrow\; \mathbf{s}_1$$

$$\mathbf{a}\,\mathbf{a}^r\,\mathbf{n}_i\mathbf{n}_i^{\ell} \;\rightarrow\; \mathbf{n}_i\mathbf{n}_i^{\ell},$$

very poor

$$\underline{(\mathbf{a}\,\mathbf{a}^{\ell})(\mathbf{a}}\,\mathbf{a}^r\,\mathbf{n}_i\mathbf{n}_i^{\ell}) \;\rightarrow\; \mathbf{n}_i\mathbf{n}_i^{\ell}.$$

$\mathbf{p}_1 =$ present participle,
$\mathbf{p}_2 =$ past participle,

sleeping dogs,
$(\underbrace{\mathbf{p}_1 \mathbf{p}_1^r}\, \mathbf{n}_2\, \underbrace{\mathbf{n}_2^\ell})\, \mathbf{n}_2 \quad \rightarrow \quad \mathbf{n}_2$

beaten dogs.
$(\underbrace{\mathbf{p}_2\, \underbrace{\mathbf{o}^\ell \mathbf{o}}\, \mathbf{p}_2^r}\, \mathbf{n}_2\, \underbrace{\mathbf{n}_2^\ell})\mathbf{n}_2 \quad \rightarrow \quad \mathbf{n}_2$

$\bar{\mathbf{j}} =$ complete infinitive with *to*.

eager to come

$$(\mathbf{a}\,\underline{\bar{\mathbf{j}}^{\ell}})\ (\underline{\bar{\mathbf{j}}}\ \underline{\mathbf{j}^{\ell}})\,\underline{\mathbf{i}} \quad \rightarrow \mathbf{a}$$

easy to dislike

$$(\mathbf{a}\,\underline{\hat{\mathbf{o}}^{\ell\ell}\,\underline{\bar{\mathbf{j}}^{\ell}})\ \underline{\bar{\mathbf{j}}}\,\underline{\mathbf{j}^{\ell}})\ (\underline{\mathbf{i}}\,\mathbf{o}^{\ell})} \quad \rightarrow \mathbf{a}$$

$$\mathbf{a} \rightarrow \overline{\mathbf{a}} \not\rightarrow \mathbf{a}.$$

too frightened

$(\overline{\mathbf{a}}\,\overline{\mathbf{a}}^{\ell})\ \mathbf{a}$ $\rightarrow$ $\overline{\mathbf{a}}$,

too frightened to flee

$(\overline{\mathbf{a}}\,\overline{\mathbf{j}}^{\ell}\ \mathbf{a}^{\ell})\ \mathbf{a}\ (\overline{\mathbf{j}}\ \mathbf{i}^{\ell})\ \mathbf{i}$ $\rightarrow$ $\overline{\mathbf{a}}$

too frightened to avoid –

$(\overline{\mathbf{a}}\,\hat{\mathbf{o}}^{\ell\ell}\,\overline{\mathbf{j}}^{\ell}\ \mathbf{a}^{\ell})\ \mathbf{a}\ (\overline{\mathbf{j}}\,\mathbf{i}^{\ell})\ (\mathbf{i}\,\mathbf{o}^{\ell})$ $\rightarrow$ $\overline{\mathbf{a}}$

**very too frightened*

$(\mathbf{a}\mathbf{a}^{\ell})\ (\overline{\mathbf{a}}\overline{\mathbf{a}}^{\ell})\ \mathbf{a}$

8. Verbs.

$$C_{jk}\ be \rightarrow \begin{pmatrix} am\ are\ is \\ was\ were\ was \end{pmatrix},$$
$$C_{jk}\ go \rightarrow \begin{pmatrix} go\ go\ goes \\ went\ went\ went \end{pmatrix},$$
$$C_{jk}\ will \rightarrow \begin{pmatrix} will\ will\ will \\ would\ would\ would \end{pmatrix}$$

$$P_j\ be \rightarrow \begin{pmatrix} being \\ been \end{pmatrix},$$
$$P_j\ go \rightarrow \begin{pmatrix} going \\ gone \end{pmatrix}$$

$$C_{13}\ go \ \rightarrow\ goes.$$
$$(\pi_3^r \mathbf{s}_1\, \underline{\mathbf{j}^\ell})\underline{\mathbf{i}} \ \rightarrow\ \pi_3^r \mathbf{s}_1$$

$$C_{2k}\ go \ \rightarrow\ went,$$
$$(\pi^r \mathbf{s}_2\, \underline{\mathbf{j}^\ell})\ \underline{\mathbf{i}} \ \rightarrow\ \pi^r \mathbf{s}_2$$

$$C_{11}\ like \ \rightarrow\ like,$$
$$(\pi_1^r \mathbf{s}_1\, \underline{\mathbf{j}^\ell})(\underline{\mathbf{i}}\, \mathbf{o}^\ell) \ \rightarrow\ \pi_1^r \mathbf{s}_1 \mathbf{o}^\ell$$

$$C_{13}\ like \ \rightarrow\ likes.$$
$$(\pi_3^r \mathbf{s}_1\, \underline{\mathbf{j}^\ell})(\underline{\mathbf{i}}\, \mathbf{o}^\ell) \ \rightarrow\ \pi_3^r \mathbf{s}_1 \mathbf{o}^\ell$$

$$I\ see\ you\ saw\ her,$$
$$\underline{\pi_1}\ \underline{(\pi_1^r}\ \mathbf{s}_1\ \underline{\mathbf{s}^\ell)}\ \underline{\pi_2}\ \underline{(\pi^r}\ \mathbf{s}_2\ \underline{\mathbf{o}^\ell)}\mathbf{o} \ \rightarrow\ \mathbf{s}_1$$

they arrive ,

$$\underline{\pi_2(\pi_2^r}\,\mathbf{s}_1\,\underline{\mathbf{j}^\ell\mathbf{i}})\quad\rightarrow\quad\mathbf{s}_1$$

they will arrive ,

$$\underline{\pi_2\ (\pi^r}\,\mathbf{s}_1\,\underline{\mathbf{j}^\ell)\ \mathbf{i}}\quad\rightarrow\quad\mathbf{s}_1$$

they have arrived ,

$$\underline{\pi_2\ (\pi_2^r}\,\mathbf{s}_1\,\underline{\mathbf{j}^\ell\mathbf{j}\,\mathbf{p}_2^\ell)(\mathbf{p}_2}\,\underline{\mathbf{j}'^\ell\mathbf{i}})\quad\rightarrow\quad\mathbf{s}_1$$

they are arriving ,

$$\underline{\pi_2\ (\pi_2^r}\,\mathbf{s}_1\,\underline{\mathbf{j}^\ell\mathbf{j}'\,\mathbf{p}_1^\ell)(\mathbf{p}_1}\,\underline{\mathbf{i}'^\ell\mathbf{i}})\quad\rightarrow\quad\mathbf{s}_1$$

they do arrive ,

$$\underline{\pi_2\ (\pi_2^r}\,\mathbf{s}_1\,\underline{\mathbf{i}^\ell)\ \mathbf{i}}\quad\rightarrow\quad\mathbf{s}_1$$

they are seen − ,

$$\underline{\pi_2(\pi_2^r}\,\mathbf{s}_1\,\underline{\mathbf{j}^\ell\mathbf{i}'\,\hat{\mathbf{o}}^{\ell\ell}\,\mathbf{p}_2^\ell)(\mathbf{p}_2}\,\underline{\mathbf{j}'^\ell\mathbf{i}\,\mathbf{o}^\ell})\quad\rightarrow\quad\mathbf{s}_1$$

they get seen − .

$$\underline{\pi_2(\pi_2^r}\,\mathbf{s}_1\,\underline{\mathbf{j}^\ell\mathbf{i}\,\hat{\mathbf{o}}^{\ell\ell}\,\mathbf{p}_2^\ell)(\mathbf{p}_2}\,\underline{\mathbf{j}'^\ell\mathbf{i}\,\mathbf{o}^\ell})\quad\rightarrow\quad\mathbf{s}_1$$

I have been arriving,

$$\pi_1(\pi_1^r\,\mathbf{s}_1\,\mathbf{j}^\ell\mathbf{j}\,\mathbf{p}_2^\ell)(\mathbf{p}_2\mathbf{j}'^\ell\mathbf{j}'\,\mathbf{p}_1^\ell)(\mathbf{p}_1\,\mathbf{i}'^\ell\mathbf{i}) \;\to\; \mathbf{s}_1$$

you will have arrived,

$$\pi_2\;(\pi^r\,\mathbf{s}_1\,\mathbf{j}^\ell)(\mathbf{j}\,\mathbf{p}_2^\ell)(\mathbf{p}_2\mathbf{j}'^\ell\mathbf{i}) \;\to\; \mathbf{s}_1$$

she will be arriving,

$$\pi_3(\pi^r\,\mathbf{s}_1\,\mathbf{j}^\ell)(\mathbf{j}'\,\mathbf{p}_1^\ell)(\mathbf{p}_1\,\mathbf{i}'^\ell\mathbf{i}) \;\to\; \mathbf{s}_1$$

we would have been arriving,

$$\pi_2(\pi^r\,\mathbf{s}_2\,\mathbf{j}^\ell)(\mathbf{j}\,\mathbf{p}_2^\ell)(\mathbf{p}_2\mathbf{j}'^\ell\mathbf{j}'\,\mathbf{p}_1^\ell)(\mathbf{p}_1\,\mathbf{i}'^\ell\mathbf{i}) \;\to\; \mathbf{s}_2$$

you would be seen −,

$$\pi_2(\pi^r\,\mathbf{s}_2\,\mathbf{j}^\ell)(\mathbf{i}'\,\hat{\mathbf{o}}^{\ell\ell}\,\mathbf{p}_2^\ell)(\mathbf{p}_2\mathbf{j}'^\ell\mathbf{i}\,\mathbf{o}^\ell) \;\to\; \mathbf{s}_2$$

$$\mathbf{a} \rightarrow \overline{\mathbf{a}}$$

she will be good ,

$$\underbrace{\pi_3\ (\pi^r}\ \mathbf{s}_1\ \underbrace{\mathbf{j}^\ell)(\mathbf{i}'\ \overline{\mathbf{a}}^\ell)}\ \underbrace{\mathbf{a}} \ \rightarrow\ \mathbf{s}_1$$

she is being good ,

$$\underbrace{\pi_3\ (\pi_3^r}\ \mathbf{s}_1\ \underbrace{\mathbf{p}_1^\ell)(\mathbf{p}_1}\ \underbrace{\mathbf{i}'^\ell \mathbf{i}'}\ \underbrace{\overline{\mathbf{a}}^\ell)}\ \underbrace{\mathbf{a}} \ \rightarrow\ \mathbf{s}_1$$

**she does be good ,*

$$\pi_3\ (\pi_3^r \mathbf{s}_1 \underbrace{\mathbf{i}^\ell)(}\ \mathbf{i}'\overline{\mathbf{a}}^\ell)\ \mathbf{a} \ \not\rightarrow\ \mathbf{s}_1$$

she is too good .

$$\underbrace{\pi_3\ (\pi_3^r}\ \mathbf{s}_1\ \underbrace{\overline{\mathbf{a}}^\ell)(\overline{\mathbf{a}}}\ \underbrace{\mathbf{a}^\ell)\mathbf{a}} \ \rightarrow\ \mathbf{s}_1$$

9. Adverbs.

I will meet you tomorrow,

$$\underline{\pi_1(\pi^r}\,\mathbf{s}_1\,\underline{\mathbf{j}^\ell)([\mathbf{i}\,\mathbf{o}^\ell)}\underline{\mathbf{o}(\mathbf{i}^r\,\mathbf{i})} \quad \to \quad \mathbf{s}_1$$

she had met him in the rain,

$$\underline{\pi_3(\pi^r}\,\mathbf{s}_2\,\mathbf{j}^\ell\underline{[\mathbf{j}\,\mathbf{p}_2^\ell)(\mathbf{p}_2}\underline{\mathbf{j}'^\ell\mathbf{i}\,\mathbf{o}^\ell)}\underline{\mathbf{o}(\mathbf{i}^r\,\mathbf{i}}\,\underline{\mathbf{o}^\ell)(\overline{\mathbf{n}}_1^\ell}\,\underline{\mathbf{n}_1)\mathbf{n}_1} \quad \to \quad \mathbf{s}_2$$

he was met when it rained.

$$\underline{\pi_3(\pi^r}\,\mathbf{s}_2\,\mathbf{j}^\ell[\mathbf{i}'\,\underline{\hat{\mathbf{o}}^{\ell\ell}}\,\underline{\mathbf{p}_2^\ell)(\mathbf{p}_2}\,\underline{\mathbf{o}^\ell)(\mathbf{i}^r}\,\mathbf{i}\,\underline{\mathbf{s}^\ell)}\,\underline{\pi_3(\pi^r}\,\underline{\mathbf{s}_2)} \quad \to \quad \mathbf{s}_2$$

$$\mathbf{p}_2\,\underline{\mathbf{j}'^\ell\mathbf{i}}\,\mathbf{o}^\ell \to \mathbf{p}_2\mathbf{o}^\ell, \qquad \pi^r\mathbf{s}_2\,\underline{\mathbf{j}^\ell\mathbf{i}} \to \pi^r\mathbf{s}_2.$$

I promised to see her today,

$$\underline{\pi_1(\pi^r}\,\mathbf{s}_2\,\underline{\mathbf{j}^\ell\mathbf{i}\,\bar{\mathbf{j}}^\ell)(\bar{\mathbf{j}}\,\mathbf{j}^\ell)([\mathbf{i}\,\underline{\mathbf{o}^\ell)\mathbf{o}(\mathbf{i}^r}\,\mathbf{i})} \;\rightarrow\; \mathbf{s}_2$$

I promised to see her today.

$$\underline{\pi_1(\pi^r}\,\mathbf{s}_2\,\underline{\mathbf{j}^\ell[\mathbf{i}\,\bar{\mathbf{j}}^\ell)(\bar{\mathbf{j}}\,\mathbf{j}^\ell)(\mathbf{i}\,\mathbf{o}^\ell)\mathbf{o}(\mathbf{i}^r}\,\mathbf{i})} \;\rightarrow\; \mathbf{s}_2$$

colourless green ideas sleep furiously.

$$(\underline{\mathbf{a}\mathbf{a}^r}\,\mathbf{n}_2\,\underline{\mathbf{n}_2^\ell)(\mathbf{a}\mathbf{a}^r}\,\mathbf{n}_2\,\underline{\mathbf{n}_2^\ell)\,\mathbf{n}_2}\,(\pi_2\,\mathbf{s}_1\,\underline{\mathbf{j}^\ell[\mathbf{i})(\mathbf{i}^r}\,\mathbf{i})} \;\rightarrow\; \mathbf{s}_1$$

under the cover

$$(\mathbf{i}^r\mathbf{i}\,\underline{\mathbf{o}^\ell)(\bar{\mathbf{n}}\,\mathbf{n}_1^\ell)\mathbf{n}_1} \;\rightarrow\; \mathbf{i}^r\mathbf{i}$$

when it rains.

$$(\mathbf{i}^r\mathbf{i}\,\underline{\mathbf{s}^\ell)\,\pi_3(\pi_3^r}\,\mathbf{s}_1) \;\rightarrow\; \mathbf{i}^r\mathbf{i}.$$

III. English Sentence Structure.

10. Negative and interrogative sentences.
11. Direct Wh-questions.
12. Indirect questions.
13. Relative clauses.

10. Negative and interrogative sentences.

The easiest way to negate a declarative sentence in English is to insert the adverb *not* in an appropriate place, namely after an auxiliary or modal verb. Thus, we assign to *not* the type xx^l, where x = **j**, **i**, $\mathbf{p}_1$, $\mathbf{p}_2$ or **a**, as illustrated by the following examples:

she would not come,
$\underbrace{\pi_3(\pi^r}\ \mathbf{s}_2\ \underbrace{\mathbf{j}^\ell)(\mathbf{j}}\ \underbrace{\mathbf{j}^\ell)}\ \underbrace{\mathbf{i}}\ \rightarrow\ \mathbf{s}_2$

she does not come,
$\underbrace{\pi_3(\pi_3^r}\ \mathbf{s}_1\ \underbrace{\mathbf{i}^\ell)(\mathbf{i}}\ \underbrace{\mathbf{i}^\ell)}\ \underbrace{\mathbf{i}}\ \rightarrow\ \mathbf{s}_1$

she is not coming,
$\underbrace{\pi_3(\pi_3^r}\ \mathbf{s}_1\ \underbrace{\mathbf{p}_1^\ell)(\mathbf{p}_1}\ \underbrace{\mathbf{p}_1^\ell)}\ \mathbf{p}_1\ \rightarrow\ \mathbf{s}_1$

she has not come,
$\underbrace{\pi_3(\pi_3^r}\ \mathbf{s}_1\ \underbrace{\mathbf{p}_2^\ell)(\mathbf{p}_2}\ \underbrace{\mathbf{p}_2^\ell)}\ \mathbf{p}_2\ \rightarrow\ \mathbf{s}_1$

she was not seen,

$$\underbrace{\pi_3(\pi_3^r\,\mathbf{s}_2\,\underbrace{\mathbf{j}^\ell\mathbf{i}'}\,\underbrace{\hat{\mathbf{o}}^{\ell\ell}\,\underbrace{\mathbf{p}_2^\ell}(\mathbf{p}_2}\,\underbrace{\mathbf{p}_2^\ell}(\mathbf{p}_2}\,\mathbf{o}^\ell)}\quad\rightarrow\quad \mathbf{s}_2$$

she is not very good,

$$\underbrace{\pi_3(\pi_3^r}\,\mathbf{s}_1\,\underbrace{\overline{\mathbf{a}}^\ell)(\overline{\mathbf{a}}}\,\underbrace{\overline{\mathbf{a}}^\ell)(\mathbf{a}}\,\underbrace{\mathbf{a}^\ell)\mathbf{a}}\quad\rightarrow\quad \mathbf{s}_1$$

she is not too good,

$$\underbrace{\pi_3(\pi_3^r}\,\mathbf{s}_1\,\underbrace{\overline{\mathbf{a}}^\ell)(\overline{\mathbf{a}}}\,\underbrace{\overline{\mathbf{a}}^\ell)(\overline{\mathbf{a}}}\,\underbrace{\overline{\mathbf{a}}^\ell)\mathbf{a}}\quad\rightarrow\quad \mathbf{s}_1$$

**she is very not good.*

$$\pi_3(\pi_3^r\mathbf{s}_1\overline{\mathbf{a}}^\ell)(\mathbf{a}\underbrace{\mathbf{a}^\ell)(\overline{\mathbf{a}}}\overline{\mathbf{a}}^\ell)\mathbf{a}\quad\not\rightarrow\quad \mathbf{s}_1$$

$$
\begin{aligned}
\textit{will not} &\rightarrow \textit{won't} \\
(\pi^r \mathbf{s}_1 \underline{\mathbf{j}^\ell})(\underline{\mathbf{j}}\,\mathbf{j}^\ell) &\rightarrow \pi^r \mathbf{s}_1 \mathbf{j}^\ell \quad ,
\end{aligned}
$$

$$
\begin{aligned}
\textit{does not} &\rightarrow \textit{doesn't} \\
(\pi^r_3 \mathbf{s}_1 \underline{\mathbf{i}^\ell})(\underline{\mathbf{i}}\,\mathbf{i}^\ell) &\rightarrow \pi^r_3 \mathbf{s}_1 \mathbf{i}^\ell \quad ,
\end{aligned}
$$

$$
\begin{aligned}
\textit{is not} &\rightarrow \textit{isn't} \\
(\pi^r_3 \mathbf{s}_1 \underline{x^\ell})(\underline{x}\,x^\ell) &\rightarrow \pi^r_3 \mathbf{s}_1 x^\ell \quad ,
\end{aligned}
$$

when $x = \mathbf{p}_2$ or $\overline{\mathbf{a}}$,

$$
\begin{aligned}
\textit{is not} &\rightarrow \textit{isn't} \\
(\pi^r_3 \mathbf{s}_1 \hat{\mathbf{o}}^{\ell\ell}\, \underline{\mathbf{p}^\ell_2})(\underline{\mathbf{p}_2}\,\mathbf{p}^\ell_2) &\rightarrow \pi^r_3 \mathbf{s}_1 \hat{\mathbf{o}}^{\ell\ell} \mathbf{p}^\ell_2 \quad .
\end{aligned}
$$

$$\mathbf{q} = \text{yes-or-no question},$$
$$\mathbf{q}_1 = \text{yes-or-no question in present tense},$$
$$\mathbf{q}_2 = \text{yes-or-no question in past tense},$$

$$\mathbf{q}_j \to \mathbf{q} \qquad (j = 1, 2)\ .$$

Metarule 1. If the finite verb form of a $_k^k$modal or auxiliary verb has the type $\pi^r \mathbf{s}_j x^l$, with $x = \mathbf{j}, \mathbf{i}_{,_k}\ \mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_2\mathbf{o}\hat{}^l$ or $\mathbf{a}$ in a declarative sentence, then it has type $\mathbf{q}_j x^l \pi^l$ in *a* question , and similarly when the subscript *j* or *k* is omitted.

does she not see me ?

$$(\mathbf{q}_1\,\underbrace{\mathbf{i}^{\ell}\,\underbrace{\pi_3^{\ell})\,\pi_3}}\,(\mathbf{i}\,\underbrace{\mathbf{i}^{\ell})(\mathbf{i}\,\underbrace{\mathbf{o}^{\ell})\mathbf{o}}}\quad\rightarrow\quad\mathbf{q}_1$$

doesn't she see me ?

$$(\mathbf{q}_1\,\underbrace{\mathbf{i}^{\ell}\,\underbrace{\pi_3^{\ell})\,\pi_3}}\,(\mathbf{i}\,\underbrace{\mathbf{o}^{\ell})\,\mathbf{o}}\quad\rightarrow\quad\mathbf{q}_1$$

am I being seen – ?

$$(\mathbf{q}_1\,\underbrace{\mathbf{p}_1^{\ell}\,\underbrace{\pi_1^{\ell})\pi_1}}\,(\mathbf{p}_1\,\underbrace{\hat{\mathbf{o}}^{\ell\ell}\,\underbrace{\mathbf{p}_2^{\ell})(\mathbf{p}_2}\,\mathbf{o}^{\ell})}\quad\rightarrow\quad\mathbf{q}_1$$

was I seen – ?

$$(\mathbf{q}_2\,\underbrace{\hat{\mathbf{o}}^{\ell\ell}\,\underbrace{\mathbf{p}_2^{\ell}\,\underbrace{\pi_1^{\ell})\,\pi_1}}\,(\mathbf{p}_2}\,\mathbf{o}^{\ell})\quad\rightarrow\quad\mathbf{q}_2$$

11. Direct Wh-questions.

she went where ?

$$\underbrace{\pi_3\,(\pi^r}\;\underbrace{\mathbf{s}_2)(\mathbf{s}^r}\;\overline{\mathbf{q}}) \;\rightarrow\; \overline{\mathbf{q}}$$

she saw whom ?

$$\underbrace{\pi_3(\pi^r}\;\underbrace{\mathbf{s}_1\mathbf{o}^\ell)(\mathbf{o}\mathbf{s}^r}\;\overline{\mathbf{q}}) \;\rightarrow\; \overline{\mathbf{q}}$$

$$\overline{\mathbf{q}} = \text{ question}$$

$$\mathbf{q} \rightarrow \overline{\mathbf{q}} \not\rightarrow \mathbf{q}.$$

where did she go ?

$(\overline{\mathbf{q}}\,\mathbf{q}^{\ell})(\mathbf{q}_2\,\mathbf{i}^{\ell}\,\pi^{\ell})\pi_3\;\mathbf{i} \;\rightarrow\; \overline{\mathbf{q}}$

**why where* did she go?

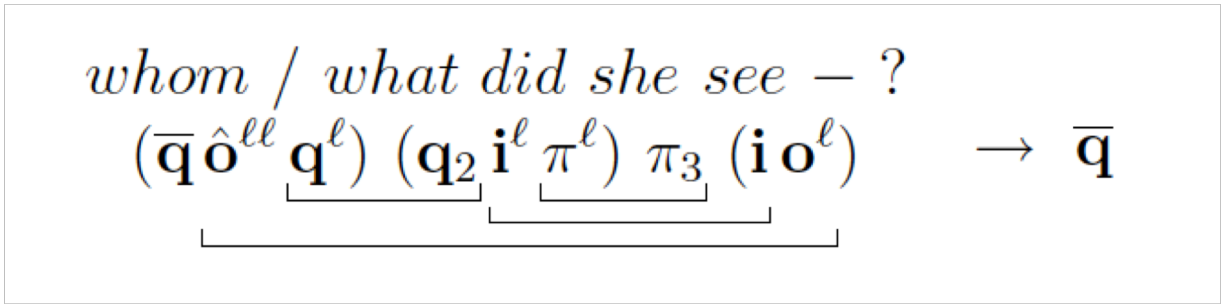
$(\overline{\mathbf{q}}\mathbf{q}^{\ell})\;(\overline{\mathbf{q}}\mathbf{q}^{\ell})$

she went to Rome.

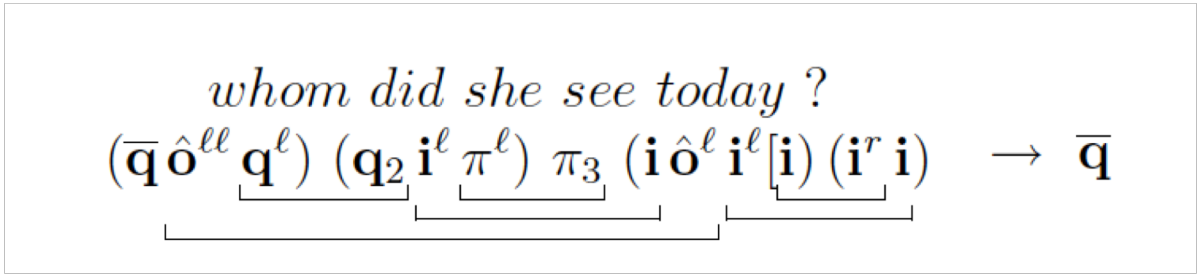
$\pi_3\,(\pi^{r}\,\mathbf{s}_2\,\mathbf{j}^{\ell}\,[\mathbf{i})\,(\mathbf{i}^{r}\,\mathbf{i}\,\mathbf{o}^{\ell})\mathbf{o} \;\rightarrow\; \mathbf{s}_2.$

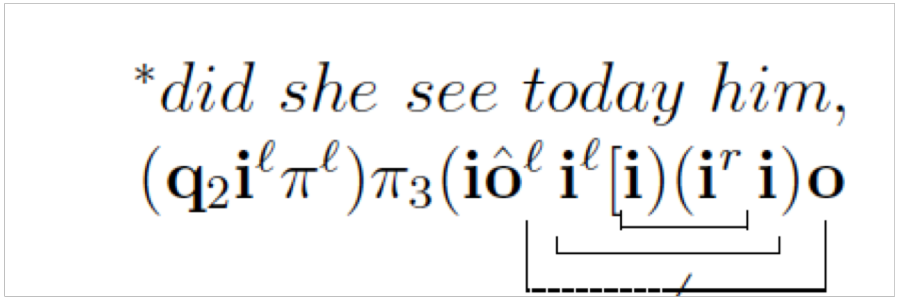
where did she go – ?

$(\overline{\mathbf{q}}\,\mathbf{i}^{\ell}\,\mathbf{i}^{\ell\ell}\,\mathbf{q}^{\ell})\,(\mathbf{q}_2\,\mathbf{i}^{\ell}\,\pi^{\ell})\,\pi_3\;\mathbf{i} \;\rightarrow\; \overline{\mathbf{q}}$



Metarule 2. In the type of a transitive verb or in a preposition (= transitive adverb), $\mathbf{o}^l$ may be replaced by $\mathbf{o}\hat{}^l\mathbf{i}^l\mathbf{i}$.





whom did she go with yesterday ?

$$(\overline{\mathbf{q}}\,\hat{\mathbf{o}}^{\ell\ell}\,\mathbf{q}^{\ell})(\mathbf{q}_2\,\mathbf{i}^{\ell}\,\pi^{\ell})\pi_3\,[\mathbf{i}\,(\mathbf{i}^{r}\,\mathbf{i}\,\hat{\mathbf{o}}^{\ell}\,\mathbf{i}^{\ell}\,[\mathbf{i})(\mathbf{i}^{r}\,\mathbf{i})\;\rightarrow\;\overline{\mathbf{q}}$$

with whom

$$(\overline{\mathbf{q}}\,\hat{\mathbf{o}}^{\ell\ell\ell}\,\overline{\mathbf{q}}^{\ell})(\overline{\mathbf{q}}\,\hat{\mathbf{o}}^{\ell\ell}\,\mathbf{q}^{\ell})\;\rightarrow\;\overline{\mathbf{q}}\mathbf{q}^{\ell}$$

whose bed did she sleep in yesterday ?

$$(\overline{\mathbf{q}}\,\hat{\mathbf{o}}^{\ell\ell}\,\mathbf{q}^{\ell}\,\mathbf{n}_1^{\ell})\,\mathbf{n}_1(\mathbf{q}_2\,\mathbf{i}^{\ell}\,\pi^{\ell})\pi_3\,[\mathbf{i}\,(\mathbf{i}^{r}\,\mathbf{i}\,\hat{\mathbf{o}}^{\ell}\,\mathbf{i}^{\ell}\,[\mathbf{i})(\mathbf{i}^{r}\,\mathbf{i})\;\rightarrow\;\overline{\mathbf{q}}$$

who came yesterday ?

$$(\overline{\mathbf{q}}\,\mathbf{s}^{\ell}\,\pi_3)(\pi_3^{r}\,\mathbf{s}_2\,\mathbf{j}^{\ell}[\mathbf{i})(\mathbf{i}^{r}\,\mathbf{i}) \;\rightarrow\; \overline{\mathbf{q}}$$

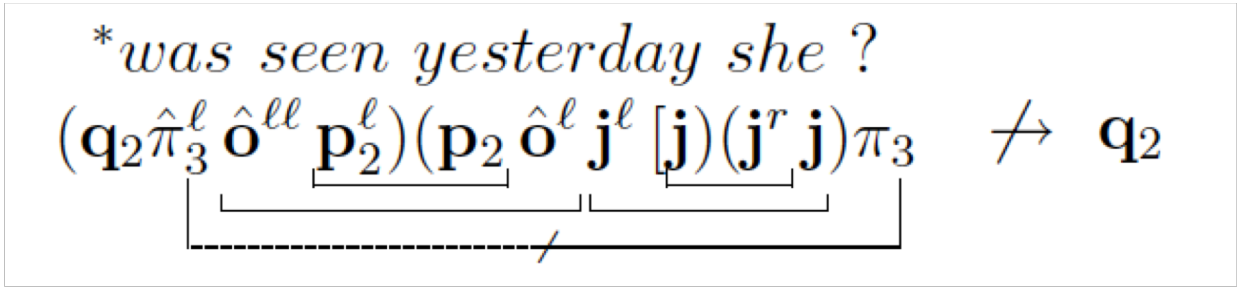
**came yesterday she ?*

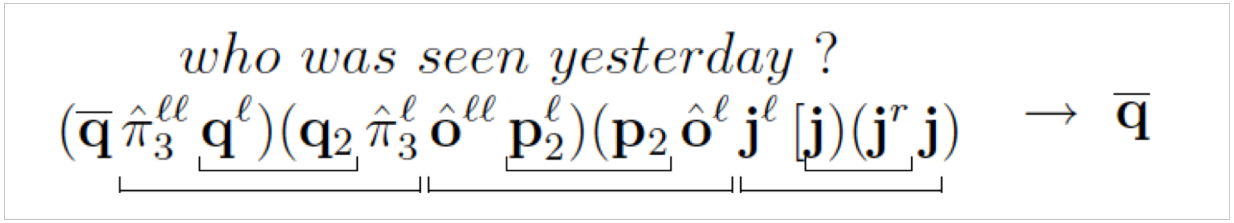
$$(\mathbf{q}_2\hat{\pi}_3^{\ell}\,\mathbf{j}^{\ell}\,[\mathbf{i})(\mathbf{i}^{r}\,\mathbf{i})\pi_3 \;\not\rightarrow\; \mathbf{q}_2$$

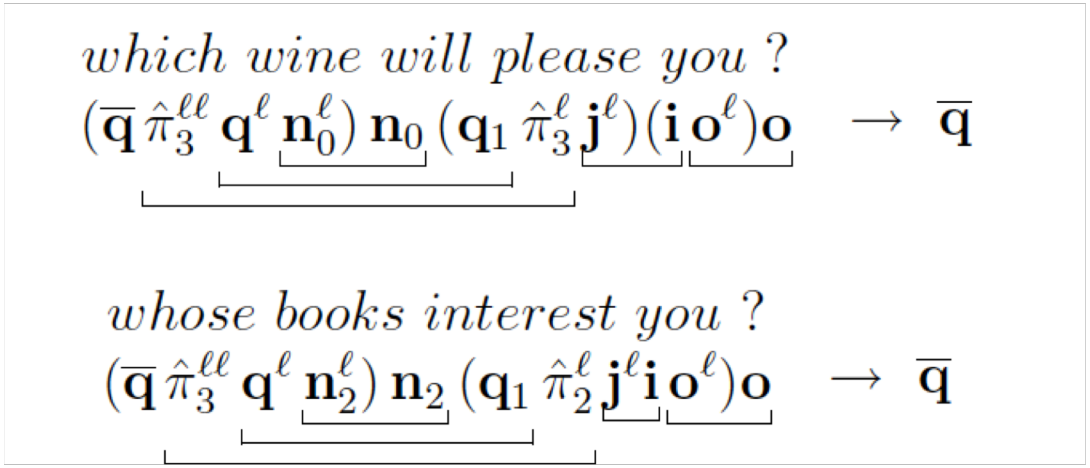
$$\hat{\pi}_3 \;\rightarrow\; \pi_3 \;\not\rightarrow\; \hat{\pi}_3.$$

who came yesterday ?

$$(\overline{\mathbf{q}}\,\hat{\pi}_3^{\ell\ell}\,\mathbf{q}^{\ell})(\mathbf{q}_2\,\pi_3^{\ell}\,\mathbf{j}^{\ell}[\mathbf{i})(\mathbf{i}^{r}\,\mathbf{i}) \;\rightarrow\; \overline{\mathbf{q}}$$


*was seen yesterday she ?
$(\mathbf{q}_2\hat{\pi}_3^{\ell}\,\hat{\mathbf{o}}^{\ell\ell}\,\mathbf{p}_2^{\ell})(\mathbf{p}_2\,\hat{\mathbf{o}}^{\ell}\,\mathbf{j}^{\ell}\,[\mathbf{j})(\mathbf{j}^{r}\,\mathbf{j})\pi_3 \not\rightarrow \mathbf{q}_2$


who was seen yesterday ?
$(\overline{\mathbf{q}}\,\hat{\pi}_3^{\ell\ell}\,\mathbf{q}^{\ell})(\mathbf{q}_2\,\hat{\pi}_3^{\ell}\,\hat{\mathbf{o}}^{\ell\ell}\,\mathbf{p}_2^{\ell})(\mathbf{p}_2\,\hat{\mathbf{o}}^{\ell}\,\mathbf{j}^{\ell}\,[\mathbf{j})(\mathbf{j}^{r}\,\mathbf{j}) \rightarrow \overline{\mathbf{q}}$


which wine will please you ?
$(\overline{\mathbf{q}}\,\hat{\pi}_3^{\ell\ell}\,\mathbf{q}^{\ell}\,\mathbf{n}_0^{\ell})\,\mathbf{n}_0\,(\mathbf{q}_1\,\hat{\pi}_3^{\ell}\,\mathbf{j}^{\ell})(\mathbf{i}\,\mathbf{o}^{\ell})\mathbf{o} \rightarrow \overline{\mathbf{q}}$
whose books interest you ?
$(\overline{\mathbf{q}}\,\hat{\pi}_3^{\ell\ell}\,\mathbf{q}^{\ell}\,\mathbf{n}_2^{\ell})\,\mathbf{n}_2\,(\mathbf{q}_1\,\hat{\pi}_2^{\ell}\,\mathbf{j}^{\ell}\mathbf{i}\,\mathbf{o}^{\ell})\mathbf{o} \rightarrow \overline{\mathbf{q}}$

12. Indirect questions.

$\mathbf{t} =$ indirect question.

I wonder whether she is coming.

$$\underbrace{\pi_1(\pi_1^r\,\mathbf{s}_1\,\underbrace{\mathbf{j}^\ell\mathbf{i}}\,\underbrace{\mathbf{t}^\ell)(\mathbf{t}}\,\underbrace{\mathbf{s}^\ell)}\,\underbrace{\pi_3(\pi_3^r\,\mathbf{s}_1\,\underbrace{\mathbf{p}_1^\ell)\mathbf{p}_1}}} \quad \to \quad \mathbf{s}_1$$

does she know if I expect her ?

$$(\mathbf{q}_1\,\underbrace{\mathbf{i}^\ell\,\underbrace{\pi_3^\ell)\pi_3}}\,\underbrace{(\mathbf{i}\,\underbrace{\mathbf{t}^\ell)(\mathbf{t}}\,\underbrace{\mathbf{s}^\ell)\,\underbrace{\pi_1(\pi_1^r\,\mathbf{s}_1\,\underbrace{\mathbf{o}^\ell)\mathbf{o}}}}} \quad \to \quad \mathbf{q}_1$$

she asked me when I would come.

$$\underbrace{\pi_3(\pi^r}\,\mathbf{s}_2\,\underbrace{\mathbf{j}^\ell\mathbf{i}\,\underbrace{\mathbf{t}^\ell\,\underbrace{\mathbf{o}^\ell)\mathbf{o}}}(\mathbf{t}\,\underbrace{\mathbf{i}^\ell\,\underbrace{\mathbf{i}^{\ell\ell}\,\underbrace{\mathbf{s}^\ell)\,\underbrace{\pi_1(\pi^r}\,\mathbf{s}_2\,\underbrace{\mathbf{j}^\ell)\mathbf{i}}}}}} \quad \to \quad \mathbf{s}_2$$

I wonder where I will go tomorrow,

$$\pi_1(\pi_1^r\,\mathbf{s}_1\,\mathbf{t}^\ell)(\mathbf{t}\,\mathbf{i}^\ell\,\mathbf{i}^{\ell\ell}\,\mathbf{s}^\ell)\,\pi_1(\pi^r\,\mathbf{s}_1\,\mathbf{j}^\ell)[\,\mathbf{i}(\mathbf{i}^r\,\mathbf{i}) \;\rightarrow\; \mathbf{s}_1$$

she doesn't know what I like,

$$\pi_3(\pi_3^r\,\mathbf{s}_1\,\mathbf{i}^\ell)(\mathbf{i}\,\mathbf{t}^\ell)(\mathbf{t}\,\hat{\mathbf{o}}^{\ell\ell}\,\mathbf{s}^\ell)\,\pi_1(\pi_1^r\,\mathbf{s}_1\,\mathbf{o}^\ell) \;\rightarrow\; \mathbf{s}_1$$

I wonder which books I should read.

$$\pi_1(\pi_1^r\,\mathbf{s}_1\,\mathbf{t}^\ell)(\mathbf{t}\,\hat{\mathbf{o}}^{\ell\ell}\,\mathbf{s}^\ell\,\mathbf{n}_2^\ell)\mathbf{n}_2\,\pi_1(\pi^r\,\mathbf{s}_2\,\mathbf{j}^\ell)(\mathbf{i}\,\mathbf{o}^\ell) \;\rightarrow\; \mathbf{s}_1$$

I ask who is coming tomorrow,

$$\pi_1(\pi_1^r\,\mathbf{s}_1\,\mathbf{t}^\ell)(\mathbf{t}\,\hat{\pi}_3^{\ell\ell}\,\mathbf{q}^\ell)(\mathbf{q}_1\,\hat{\pi}_3^\ell\,\mathbf{p}_1^\ell)(\mathbf{p}_1\,\mathbf{i}'^\ell\,[\mathbf{i})(\mathbf{i}^r\,\mathbf{i}) \;\rightarrow\; \mathbf{s}_1$$

I know which girl likes you.

$$\pi_1(\pi_1^r\,\mathbf{s}_1\,\mathbf{t}^\ell)(\mathbf{t}\,\hat{\pi}_3^{\ell\ell}\,\mathbf{q}^\ell\,\mathbf{n}_1)\mathbf{n}_1(\mathbf{q}_1\,\hat{\pi}_3^\ell\,\mathbf{o}^\ell)\mathbf{o} \;\rightarrow\; \mathbf{s}_1$$

I wonder where to go,

$$\underbrace{\pi_1(\pi_1^r}\,\mathbf{s}_1\,\underbrace{\mathbf{t}^\ell)(\mathbf{t}}\,\underbrace{\bar{\mathbf{j}}^\ell)(\bar{\mathbf{j}}}\,\underbrace{\mathbf{i}^\ell)\mathbf{i}} \quad \rightarrow \quad \mathbf{s}_1$$

I wonder whom to see tomorrow.

$$\underbrace{\pi_1(\pi_1^r}\,\mathbf{s}_1\,\underbrace{\mathbf{t}^\ell)(\mathbf{t}}\,\underbrace{\hat{\mathbf{o}}^{\ell\ell}\,\bar{\mathbf{j}}^\ell)(\bar{\mathbf{j}}}\,\underbrace{\mathbf{i}^\ell)(\mathbf{i}}\,\underbrace{\hat{\mathbf{o}}^\ell\,\mathbf{j}^\ell\,[\mathbf{i})(\mathbf{i}^r}\,\underbrace{\mathbf{i})}} \quad \rightarrow \quad \mathbf{s}_1$$

I know which books to read.

$$\underbrace{\pi_1(\pi_1^r}\,\mathbf{s}_1\,\underbrace{\mathbf{t}^\ell)(\mathbf{t}}\,\underbrace{\hat{\mathbf{o}}^{\ell\ell}\,\bar{\mathbf{j}}^\ell\,\mathbf{n}_2^\ell)\mathbf{n}_2(\bar{\mathbf{j}}}\,\underbrace{\mathbf{i}^\ell)(\mathbf{i}}\,\underbrace{\mathbf{o}^\ell)} \quad \rightarrow \quad \mathbf{s}_1$$

IV. Other verbal complements.

14. Doubly transitive verbs.
15. Sentential complements
16. Discontinuous dependencies.
17. Quasi-sentences.

V. Unbounded Dependencies and Constraints.

18. Unbounded dependencies.
19. Unblocking maneuver.
20. Coordinate and subordinate conjunctions.
21. Summary of constraints.

VI. Semantic Considerations.

22. Quantifiers.
23. Anaphora.
24. Kinship grammar.
25. Capulet versus Montague.

VII. Background and limitation of pregroup grammars.

26. Remarks on the history of categorial grammar.
27. The mathematical background.
28. Limitation of free pregroup grammars.
29. Conclusion.

VIII. Summary.

30. Mathematical Rules.
31. The poset of basic types.
32. Type assignments.
33. Dictionary rewriting rules.

IX. Appendices.

34. Related mathematical recreations.
35. Selected etymologies.
36. Postscript on Minimalism.
37. Acknowledgements.

Deep parsing in Watson

「Watsonでの深いパージング」

M. C. McCord, J. W. Murdock, and B. K. Boguraev

http://goo.gl/VLuWKc

"Deep parsing in Watson”

- この論文の著者の一人 M. C. McCord は、これまで紹介してきた Slot Grammarの創始者である。
- 「文章を深く解析する二つのコンポーネント、英文スロット文法パーザー（English Slot Grammar (ESG) parser）と述語引数構造ビルダー（predicate-argument structure (PAS) builder）は、IBM Watsonで利用される質問と文字情報の双方に、答えを見つけ仮説を提供する、核心的な言語分析の能力を提供している。特に、これらのコンポーネントは、質問の分析・回答候補の生成・文中の証拠の分析に基本的な役割を果たしている。

http://goo.gl/VLuWKc

"Deep parsing in Watson”

- Watsonプロジェクトの一部として、ESGは機能拡張され、Jeopardy!の質問や、あらかじめ用意された参照データに対するそのパフォーマンスは改善された。PASは、ESGの上に、より高度な分析を支援するために構築された。この論文では、これらのコンポーネントについて述べ、それらがWatsonのパターン・ベースの関係抽出コンポーネントの中でどのように利用されているかを示そうと思う。また、ESGパージングのコンポーネント・レベルでのパフォーマンス評価の定量的な結果も提供したい。」

http://goo.gl/VLuWKc

PAS（predicate-argument structure）ビルダー

- 次のような文を考えよう。
  - I heard that Edison invented the phonograph in 1877.
  - I heard that Edison invented a phonograph in 1877.
- この二つの文は、違った意味を持つ。そのことは、ESGでの解析では正確に反映される。ただ、PASでは、この区別は捨象される。

- 次の例でも、EGBでは異なる解析が得られるが、PASでは同じである。
  - Odysseus was bold and clever.
  - bold and clever Odysseus.

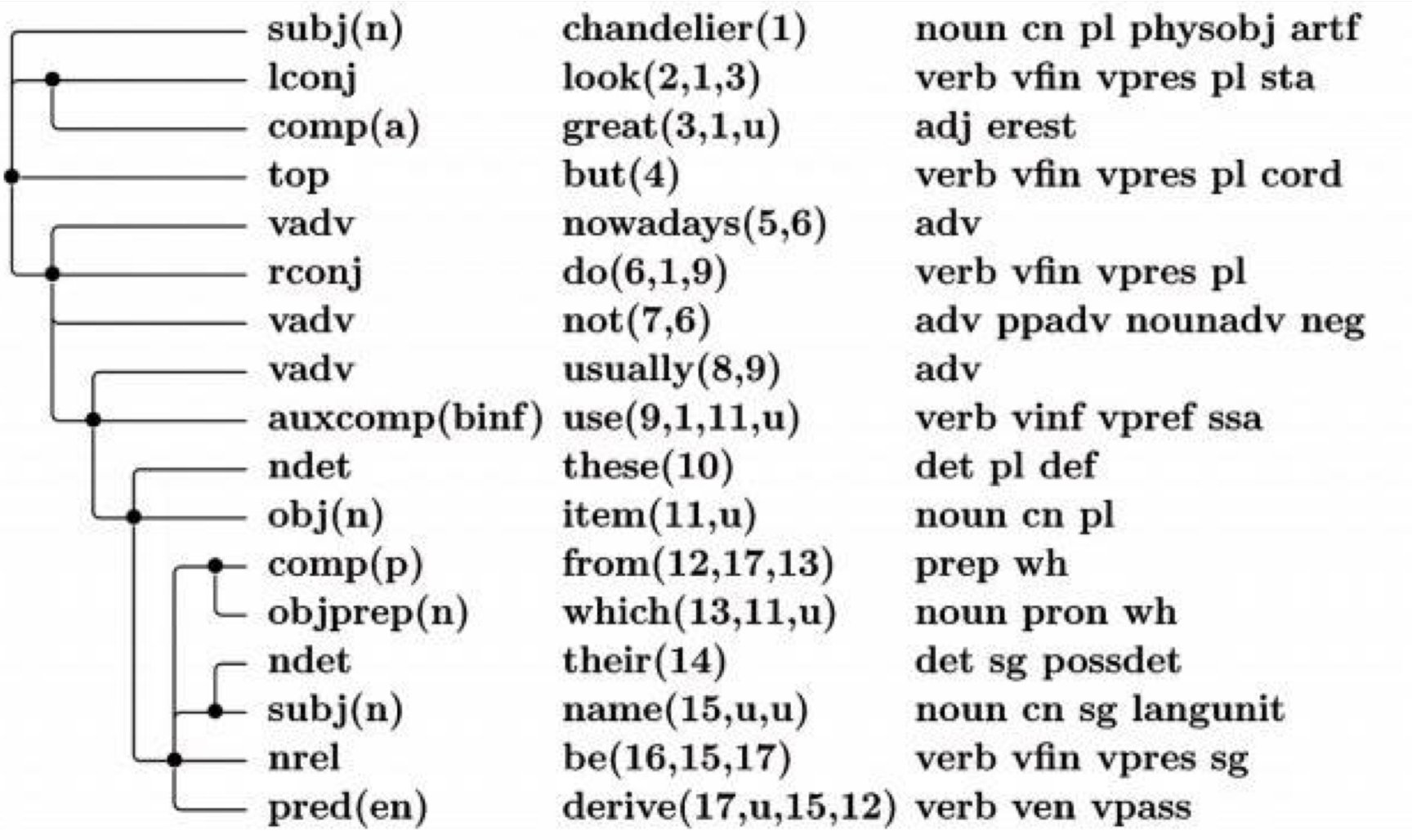
ESGパーザーとPASビルダー

- このように、PASビルダーは、ESGパーザーの解析結果の細部を省略して、単純化・抽象化する。これによって、失われる「意味」や「ニュアンス」もあるのだが、こうした単純化した形式で意味を抽出したほうが、大量のデータとして扱うには効率的である。

- 次の図は、Jeopardyの問題
  “Chandeliers look great but nowadays do not usually use these items from which their name is derived.”
  を、ESGで解析したものである。

"Chandeliers look great but nowadays do not usually use these items from which their name is derived."

ESGでの解析



subj(n)	chandelier(1)	noun cn pl physobj artf
lconj	look(2,1,3)	verb vfin vpres pl sta
comp(a)	great(3,1,u)	adj erest
top	but(4)	verb vfin vpres pl cord
vadv	nowadays(5,6)	adv
rconj	do(6,1,9)	verb vfin vpres pl
vadv	not(7,6)	adv ppadv nounadv neg
vadv	usually(8,9)	adv
auxcomp(binf)	use(9,1,11,u)	verb vinf vpref ssa
ndet	these(10)	det pl def
obj(n)	item(11,u)	noun cn pl
comp(p)	from(12,17,13)	prep wh
objprep(n)	which(13,11,u)	noun pron wh
ndet	their(14)	det sg possdet
subj(n)	name(15,u,u)	noun cn sg langunit
nrel	be(16,15,17)	verb vfin vpres sg
pred(en)	derive(17,u,15,12)	verb ven vpass

"Chandeliers look great but nowadays do not usually use these items from which their name is derived."

PASでの解析

```
chandelier (1)
look (2, subj:1, comp:3)
great (3)
but (4, lconj:2, rconj:9) [ top predicate]
nowadays (5)
not (7)
usually (8)
use (9, subj:1, obj:11, vadv:5, vadv:7,
  vadv:8)
item (11, nrel:17) [ determiner: these]
from (12, objprep:13)
which (13)
their (14)
name (15, ndet:14) [ determiner: their]
derive (17, obj:15, comp:12)
```

PASでの解析結果はESGでの解析結果より、シンプルである。ただ、述語と引数の関係を捉えることで、意味の深い理解に進んでいる。

Automatic knowledge extraction from documents

「ドキュメントからの自動知識抽出」

J. Fan, A. Kalyanpur, D. C. Gondek, and D. A. Ferrucci

http://goo.gl/MexfSl

ドキュメントからの知識の抽出

- Watsonでは、質問（クイズで言えば「ヒント」）の文解析と、大量のドキュメント・データから「知識」を抽出する文解析とでは、多少異なる手法が利用されている。
- 質問（ヒント）の解析には、これまで見たきたように、「英文スロット文法パーザー（English Slot Grammar (ESG) parser）」と「述語引数構造ビルダー（predicate-argument structure (PAS) builder）」を使うのだが、ドキュメントから「知識」を抽出するのに利用されるシステムは、"PRISMATIC".と呼ばれている。
- Watson論文集中の、次の論文がこの問題に当てられている。"Automatic knowledge extraction from documents「文書からの自動的な知識の抽出」" http://goo.gl/MexfSl

ドキュメントからの知識の抽出

- 「IBM WatsonのようなDeepQAシステムにとって、どんな領域の話題にも対応して質問に答えることに成功する為には、非常に膨大な知識にアクセスすることが、本質的に重要である。
- 形式化された知識の表現は、それによって推論が容易になるという利点があるのだが、オープンな領域で構造化されていないデータから構造化された知識を獲得することは、しばしば困難であり、コストがかかりすぎる。
- 我々の中心的な仮説は、浅い文法的な知識とそれが含意する意味は、容易に獲得でき、かつ、質問応答システムの多くの領域で利用できるということである。

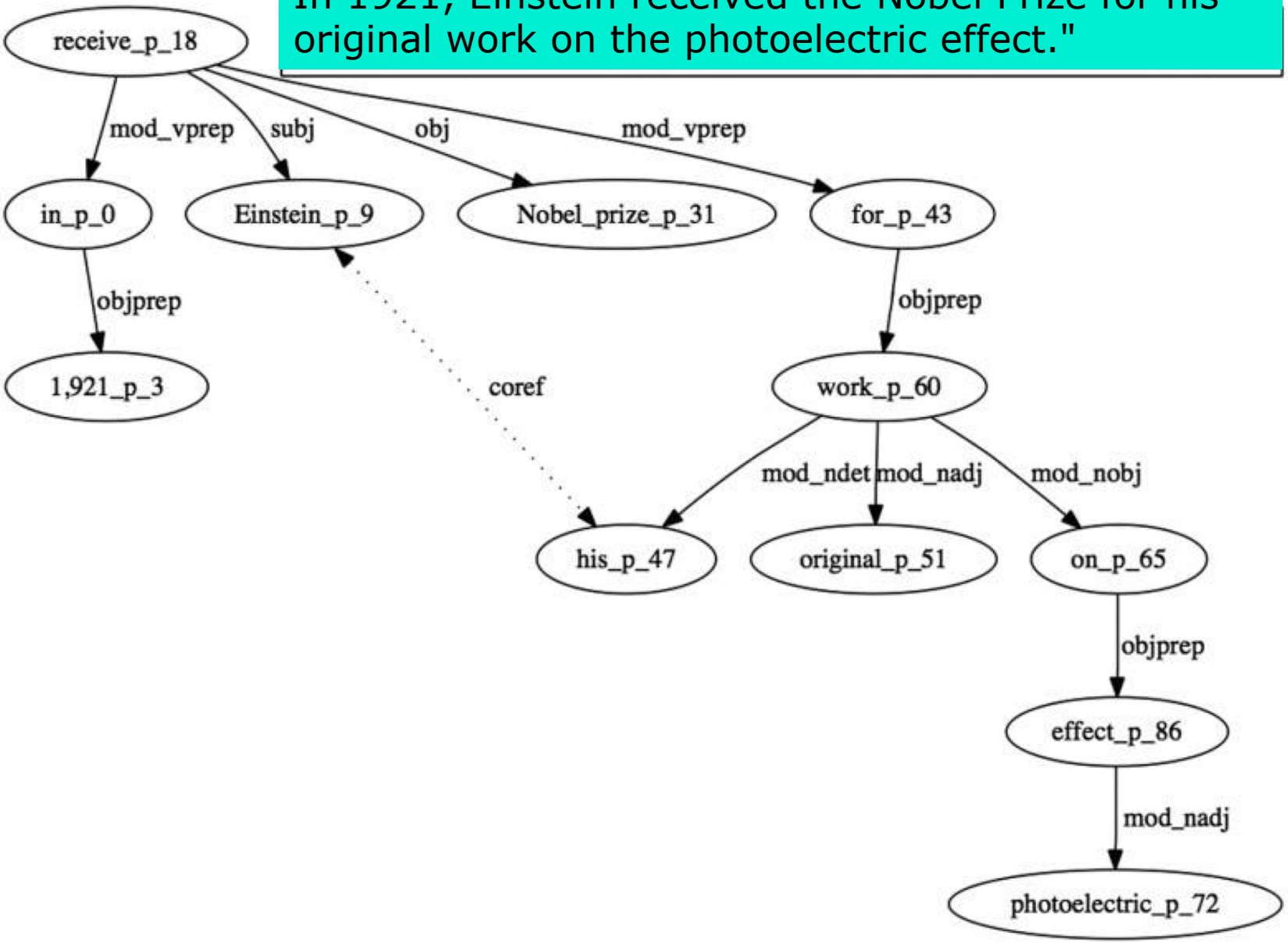
ドキュメントからの知識の抽出

- 我々は、文法的な知識とそれが含意する意味の抽出を二段階で行う。第一に、大規模なドキュメントの集まりからの浅い知識が、自動的に抽出される。第二に、自動的に抽出された浅い知識から集約された統計情報から、追加的な意味を推論する。
- この論文では、どのような浅い知識が抽出され、それがどのように大規模なコーパスから自動的になされるのか、そして、集約された統計情報から、どのようにして追加的な意味が推論されるのかを詳述する。同時に、知識抽出の様々な方法がIBMのDeepQAシステム全体の中で、どのように利用されているかについて簡単に述べたい。」

PRISMATIC

- PRISMATICについては、内容が重なっているが、次の論文がある。"PRISMATIC: Inducing Knowledge from a Large Scale Lexicalized Relation Resource" http://goo.gl/AsLr94
- PRISMATICが、次の文章 "In 1921, Einstein received the Nobel Prize for his original work on the photoelectric effect."を解析した様子を最初の図に示す。ベースになっているのは、Slot Grammer だが、スロットの種類が ESGパーザーとは若干異なっている。ESGパーザーのスロットと基本的なFeatureが組み合わせてスロットを拡張したように見える。

In 1921, Einstein received the Nobel Prize for his original work on the photoelectric effect."

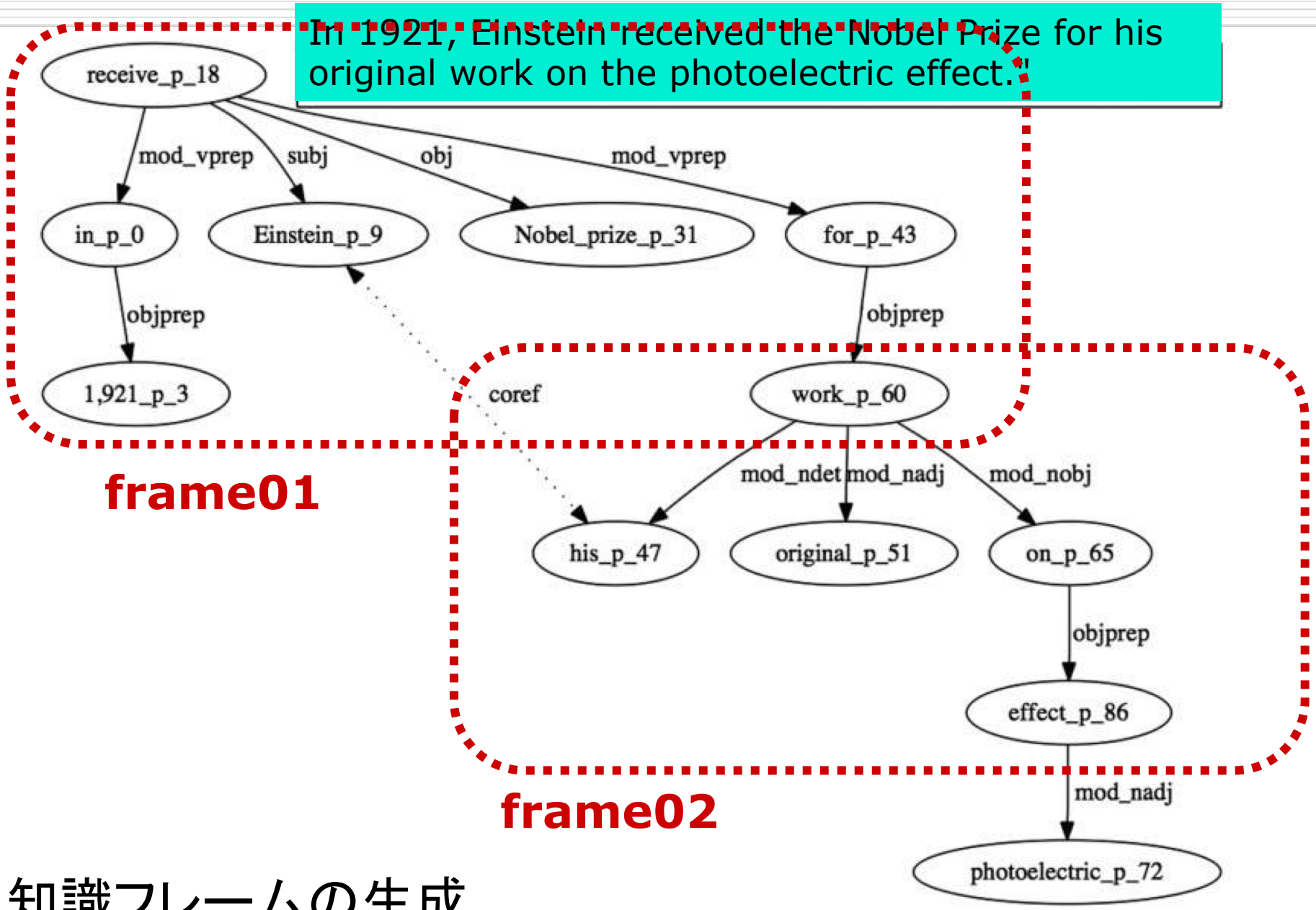


PRISMATIC

- また、以前に見た「質問」の解析では、ESGパーザーでの解析に続く「述語引数構造（predicate-argument structure (PAS) 」の抽出が目標だったのだが、ここではそうした二段階の処理は行われていない。

知識フレームの生成

- 二つ目の図は、元の文から、“In 1921, Einstein received the Nobel Prize for Frame02”というFrame01と、“his original work on effect.” というFrame02の二つのフレームが作られることを表す。（“photoelectric”が、切り捨てられてしまっているが。）


In 1921, Einstein received the Nobel Prize for his original work on the photoelectric effect.
receive_p_18
mod_vprep
subj
obj
mod_vprep
in_p_0
Einstein_p_9
Nobel_prize_p_31
for_p_43
objprep
objprep
1,921_p_3
coref
work_p_60
frame01
mod_ndet
mod_nadj
mod_nobj
his_p_47
original_p_51
on_p_65
objprep
effect_p_86
mod_nadj
frame02
photoelectric_p_72

知識フレームの生成

PRISMATICが生成した知識フレーム

Frame01	
verb	receive
subj	Einstein
type	PERSON/SCIENTIST
obj	Nobel prize
mod_vprep	in
objprep	1921
type	YEAR
mod_vprep	for
objprep	Frame02

Frame02	
noun	work
mod_ndet	his/Einstein
mod_nobj	on
objprep	effect

膨大な知識フレームの生成

- PRISMATICは、Wikipediaを始めとする30GBのドキュメント・データから、こうして、10億個もの知識フレームを生成しているという。すごい。「質問」の解析は、リアルタイムになされなければいけないのだが、こうした知識フレームの生成は、もちろん、事前に行われている。

知識フレームと統計情報

- この膨大な知識フレームの集積から、どのように統計情報が引き出されるかを見てみよう。
- 例えば、"Napoleon annexed Piedmont in 1859「ナポレオンは、1859年に、ピーモンドを併合した」" という文章から生成されるフレームを考えてみよう。基本的には、S-V-Oの形のフレームなのだが、こうした形で、動詞 "annex"（「併合する」）を含む多数のフレームを集めると、annexは、「地域」を型に持つ目的語を取ることが多いことが、統計的にわかる。

知識フレームと統計情報

- 今度は、次のような文を考えよう。"He ordered a Napoleon at the restaurant." ここでのNapoleonは、もちろん、先の文章でのNapoleonとは違うのだが、そのことは、次のようにして推論できる。
- “Napoleon”の型が分からなかったら、膨大な知識フレームの中から、この文章から生成されるフレームと同じ（ただし“Napoleon”の部分の型は「不明」として）次のような形をしているフレームを検索する。そうすると、このorderの目的語の型は、統計的には「食べ物」だということがわかる。

統計情報をLATの推定に利用

{
<subject type,"Person">
<verb,"order">
<object type,"**?**"> <== **"Napoleon"の型は？**
<verb prep,"at">
<prep,"restaurant">
}

- 質問応答システムでは、質問が要求している答えの文法的な型（これを LAT “Lexical Answer Type” という）を推定することは、とても重要になるのだが、ここでも、この統計的な情報は役にたつ。

Question analysis:
How Watson reads a clue

「質問の分析：Watsonは、どのようにヒントを読むのか？」

A. Lally, J. M. Prager, M. C. McCord, B. K. Boguraev, S. Patwardhan, J. Fan, P. Fodor, and J. Chu-Carroll

http://brenocon.com/watson_special_issue/02%20question%20analysis.pdf

Watsonの構文解析と意味解析の能力

- IBM Watsonシステムの処理の最初のステージは、何が求められているのか、そして、それに答えるにはどのようなアプローチが最良なのかを決定するために、質問の詳細な分析を行うことである。質問の分析は、Watsonの構文解析と意味解析の次のような能力を利用する。深いSlot Grammarのパーザー、名前を持つエンティティの認識、参照を解決するコンポーネント、関係を抽出するコンポーネントである。
- 我々は、この分析から得られる諸特徴を用いて、質問中の本質的に重要な要素を検出するために、様々な検出ルールとクラス分けの手法を適用する。質問中の重要な要素としては、次のものがある。

POETS & POETRY: He was a bank clerk in the Yukon before he published “Songs of a Sourdough” in 1907.

- **Focus** は、質問の一部で、答えを指しているものである。この上のサンプルでは、Focusは、”He” である。
- **LATs** （Lexical Answe Type）は、質問中の語句で、どのような型のエンティティが質問されているかを示すものである。Focusの主辞（見出語）は、一般的にはLATである。しかし質問には、そのほかにもLATを含むことがしばしばある。Jeopardy!の領域では、質問のカテゴリーが、こうした追加的なLATの源になる。この例では、“he”, “clerk”, “poet” がLATである。

POETS & POETRY: He was a bank clerk in the Yukon before he published “Songs of a Sourdough” in 1907.

- **Question Classification** は、質問を、一つあるいは複数の、もっと広い型に属するものとみなす。この例では、この質問は、最も一般的なクラス、Factroidに属する。
- **QSections** は、その解釈が特別な処理を必要とする、質問の断片である。QSectionの最も重要な利用は、一つは、答えの構文上の制約条件（例えば、「4文字」とか「3語」といった）であり、もう一つは、質問をいくつかの下位の質問に分解することである。

focus と LAT 検出の基本

- 限定詞 “this” あるいは “these”を伴った名詞句:
  - THEATRE: A new play based on *this Sir Arthur Conan Doyle canine classic* opened on the London stage in 2007.
- 代名詞としての This あるいは these ：
  - ‘88: In April 1988, Northwest became the first U.S. air carrier to ban *this* on all domestic flights.
- 質問が名詞句だった場合、便宜的に、質問全体に focus のラベルをつける。
  - AMERICAN LIT: *Number of poems Emily Dickinson gave permission to publish during her lifetime.*

focus の検出

- 次の代名詞のいずれか “he/she/his/her/him/hers”:
  - OUT WEST: *She* joined Buffalo Bill Cody’s Wild West Show after meeting him at the Cotton Expo in New Orleans.
- 次の代名詞のいずれか “it/they/them/its/their”:
  - ME “FIRST”!: *It* forbids Congress from interfering with a citizen’s freedom of religion, speech, assembly, or petition.
- 代名詞 “one”:
  - 12-LETTER WORDS: Leavenworth, established in 1895, is a federal *one*.
- 上のいずれも適用されなかったら、質問は、focusがないかもしれない。

LAT の検出

- もし focusが接続詞だったら、その要素を抽出する
  - HENRY VIII: Henry destroyed the Canterbury Cathedral Tomb of *this **saint** and **chancellor** of Henry II.*
- “(Focus) of X”. (Focus)が、次のどれかだったら one/name/type/kind LAT X を抽出する:
  - HERE, PIGGY, PIGGY, PIGGY: Many a mom has compared her kid’s messy room to *this kind of hog **enclosure**.*

LAT の検出

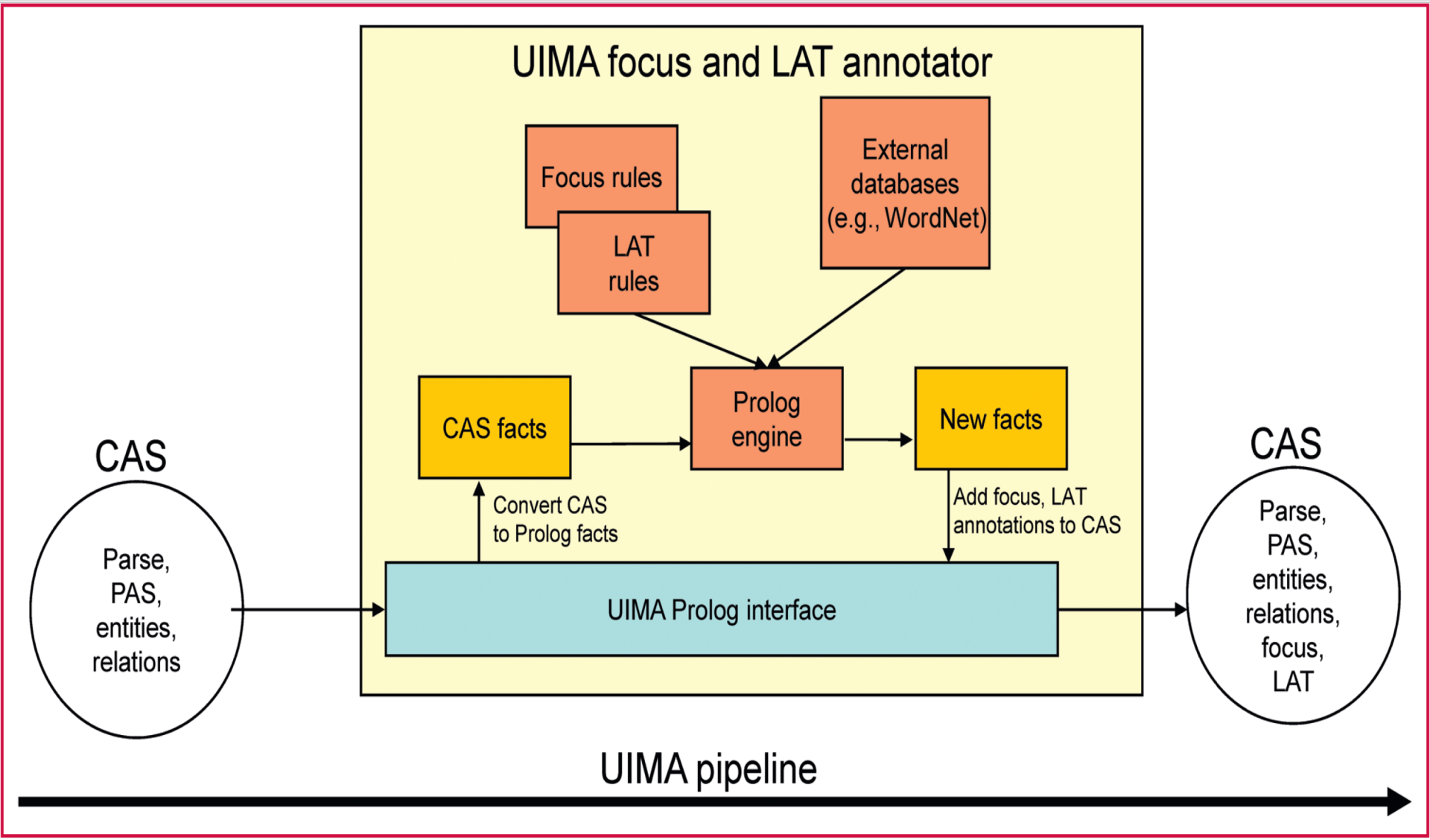
- “(Focus) for X”. (Focus)が、次のどれかだったらone/name/type/kind LAT X を抽出する:
  - COMPANY NAME ORIGINS: James Church chose *this name for his* ***product*** because the symbols of the god Vulcan represented power.
- focusが検出されず、カテゴリーが名詞句だったら、カテゴリーの見出しを、LATとする。
  - HEAVY METAL BANDS: “Seek & Destroy”, “Nothing Else Matters”, “Enter Sandman”.

POETS & POETRY: He was a bank clerk in the Yukon before he published "Songs of a Sourdough" in 1907.

- この質問の例では、構文と意味の解析は、多くの中でも、**publish(e1, he, "Songs of a Sourdough")** と **in(e2, e1, 1907)**というに述語に注目する。後者は、発行のイベント e1 が1907年に起きたことを述べている。
- 同時に解析は、二つの"he"と"clerk"が、共通参照であることを見出し、"Yukon"は地政学的エンティティであり、"Songs of a Sourdough"は、作品であると同定する。こうして、**authorOf(focus, "Songs of a Sourdough")** と **temporalLink (publish (. . .), 1907)** といった関係を抽出する。

ルールのPrologでの実装

- Watsonの中では、たいてい、質問の分析は、UIMA(Unstructured Information Management Architecture)を使った、コンポーネントを組み合わせたパイプラインとして実装されている。
- Watsonのプロジェクトの質問分析の仕事の大部分は、PASとWodNetのような様々な外部のデータベース上のルールとして実装されている。



In UIMA, the common analysis structure (CAS) is a dynamic data structure that contains unstructured data (i.e., data whose intended meaning is still to be inferred) and structured information inferred from this data, encoded as feature structures.

CAS(Common Analysis Structure)のPrologへの翻訳

- それぞれのCAS の特徴構造は、ユニークな整数値のIDを割り振られる。特徴構造のそれぞれの特徴（プロパティ）は、**feature_name(id, value)**という形で、factとなる。
- もし、featureの値が、他の特徴構造であった場合には、対象の特徴構造のIDが、Prologのfactの値のスロットに使われる。
- Prolog ルールから導かれる、いかなる新しいfactも、このCASが新しい特徴構造であることを示すものとして主張される。そして、現在のUIMAパイプラインの次のAnnotatorに渡される。