

Artificial Intelligence for Text Analytics

Question Answering and Dialogue Systems

1102AITA09

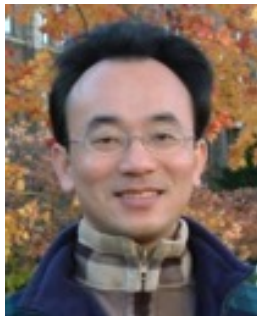
MBA, IM, NTPU (M5026) (Spring 2022)

Tue 2, 3, 4 (9:10-12:00) (B8F40)



<https://meet.google.com/paj-zhji-mya>

aws
educate | Cloud
Ambassador
2020 Cohort



Min-Yuh Day, Ph.D,
Associate Professor

Institute of Information Management, National Taipei University

<https://web.ntpu.edu.tw/~myday>



Syllabus

Week	Date	Subject/Topics
1	2022/02/22	Introduction to Artificial Intelligence for Text Analytics
2	2022/03/01	Foundations of Text Analytics: Natural Language Processing (NLP)
3	2022/03/08	Python for Natural Language Processing
4	2022/03/15	Natural Language Processing with Transformers
5	2022/03/22	Case Study on Artificial Intelligence for Text Analytics I
6	2022/03/29	Text Classification and Sentiment Analysis

Syllabus

Week	Date	Subject/Topics
7	2022/04/05	Tomb-Sweeping Day (Holiday, No Classes)
8	2022/04/12	Midterm Project Report
9	2022/04/19	Multilingual Named Entity Recognition (NER), Text Similarity and Clustering
10	2022/04/26	Text Summarization and Topic Models
11	2022/05/03	Text Generation
12	2022/05/10	Case Study on Artificial Intelligence for Text Analytics II

Syllabus

Week Date Subject/Topics

13 2022/05/17 Question Answering and Dialogue Systems

**14 2022/05/24 Deep Learning, Transfer Learning,
Zero-Shot, and Few-Shot Learning for Text Analytics**

15 2022/05/31 Final Project Report I

16 2022/06/07 Final Project Report II

17 2022/06/14 Self-learning

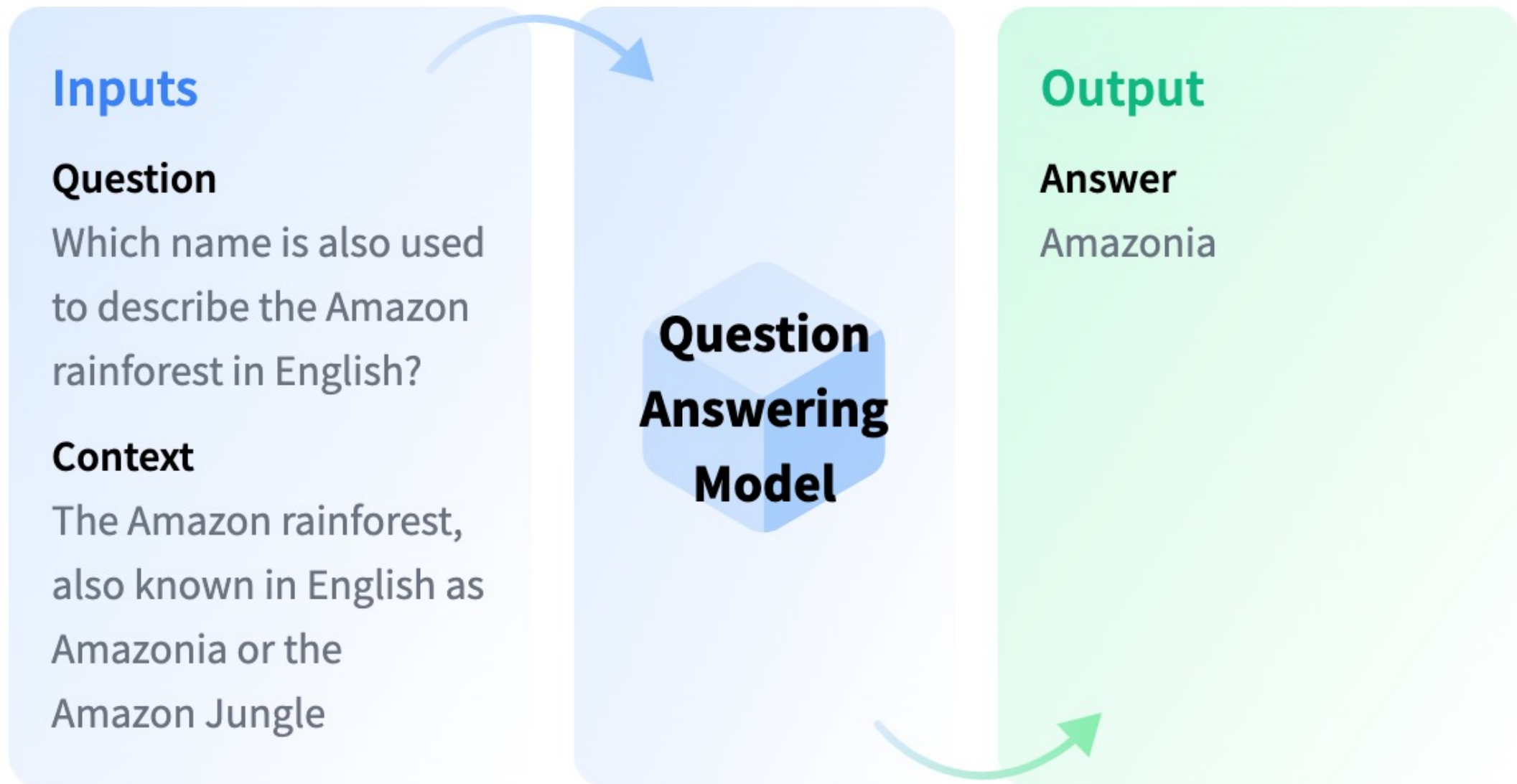
18 2022/06/21 Self-learning

Question Answering and Dialogue Systems

Outline

- **Question Answering**
- **Dialogue Systems**
- **Task Oriented Dialogue System**

Question Answering



Question Answering

⚡ Question Answering demo

using [deepset/roberta-base-squad2](#)

☰ Question Answering

Example 2



Where do I live?

Compute

Context

My name is Michael and I live in Taipei.

Computation time on cpu: 0.0492 s

Taipei

0.920

</> JSON Output

🖥️ Maximize

Question Answering

```
!pip install transformers
from transformers import pipeline
qamodel = pipeline("question-answering")
question = "Where do I live?"
context = "My name is Michael and I live in Taipei."
qamodel(question = question, context = context)
```

```
{'answer': 'Taipei', 'end': 39, 'score': 0.9730741381645203, 'start': 33}
```

Question Answering

```
from transformers import pipeline
qamodel = pipeline("question-answering", model='deepset/roberta-base-squad2')
question = "Where do I live?"
context = "My name is Michael and I live in Taipei."
output = qamodel(question = question, context = context)
print(output[ 'answer' ])
```

Taipei

Tamkang University

淡江大學

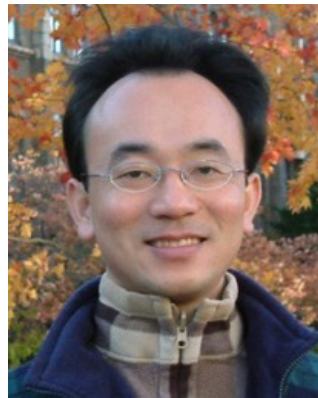
2011



Tamkang
University

**IMTKU Textual Entailment System for
Recognizing Inference in Text
at NTCIR-9 RITE**

**Department of Information Management
Tamkang University, Taiwan**



Min-Yuh Day

myday@mail.tku.edu.tw

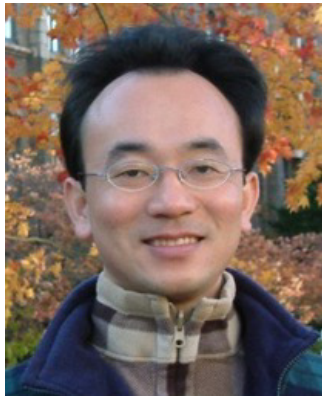


Chun Tu

NTCIR-9 Workshop, December 6-9, 2011, Tokyo, Japan

IMTKU Textual Entailment System for Recognizing Inference in Text at **NTCIR-10** RITE-2

Department of Information Management
Tamkang University, Taiwan



Min-Yuh Day



Chun Tu



Hou-Cheng Vong



Shih-Wei Wu



Shih-Jhen Huang

myday@mail.tku.edu.tw

IMTKU Textual Entailment System for
Recognizing Inference in Text at NTCIR-11 RITE-VAL

Tamkang University

淡江大學

2014



Min-Yuh Day



Ya-Jung Wang



Che-Wei Hsu



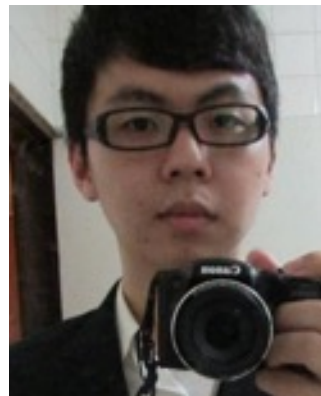
En-Chun Tu



Huai-Wen Hsu



Yu-An Lin



Shang-Yu Wu



Yu-Hsuan Tai



Cheng-Chia Tsai

IMTKU Question Answering System for World History Exams at NTCIR-12 QA Lab2

Department of Information Management
Tamkang University, Taiwan

Sagacity Technology



Min-Yuh Day



Cheng-Chia Tsai



Wei-Chun Chung



Hsiu-Yuan Chang



Tzu-Jui Sun



Yuan-Jie Tsai



Jin-Kun Lin



Cheng-Hung Lee



Yu-Ming Guo



Yue-Da Lin



Wei-Ming Chen



Yun-Da Tsai



Cheng-Jih Han



Yi-Jing Lin



Yi-Heng Chiang



Ching-Yuan Chien

myday@mail.tku.edu.tw



IMTKU Question Answering System for World History Exams at **NTCIR-13** QALab-3

Department of Information Management
Tamkang University, Taiwan



Min-Yuh Day



Chao-Yu Chen



Wanchu Huang



Shi-Ya Zheng



I-Hsuan Huang



Tz-Rung Chen



Min-Chun Kuo



Yue-Da Lin



Yi-Jing Lin

myday@mail.tku.edu.tw



IMTKU Emotional Dialogue System for Short Text Conversation at **NTCIR-14** STC-3 (CECG) Task

Department of Information Management
Tamkang University, Taiwan



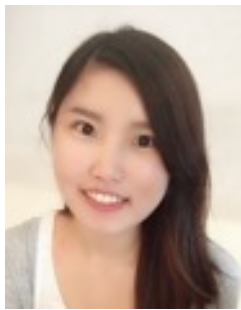
Min-Yuh Day



Chi-Sheng Hung



Yi-Jun Xie



Jhih-Yi Chen



Yu-Ling Kuo



Jian-Ting Lin



IMTKU Multi-Turn Dialogue System Evaluation at the NTCIR-15 DialEval-1 Dialogue Quality and Nugget Detection

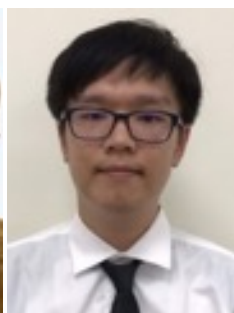
¹ Zeals Co., Ltd. Tokyo, Japan

² Information Management, Tamkang University, Taiwan

³ Information Management, National Taipei University, Taiwan



Mike Tian-Jian Jiang¹



Zhao-Xian Gu²



Cheng-Jhe Chiang²



Yueh-Chia Wu²



Yu-Chen Huang²



Cheng-Han Chiu²



Sheng-Ru Shaw²



Min-Yuh Day³

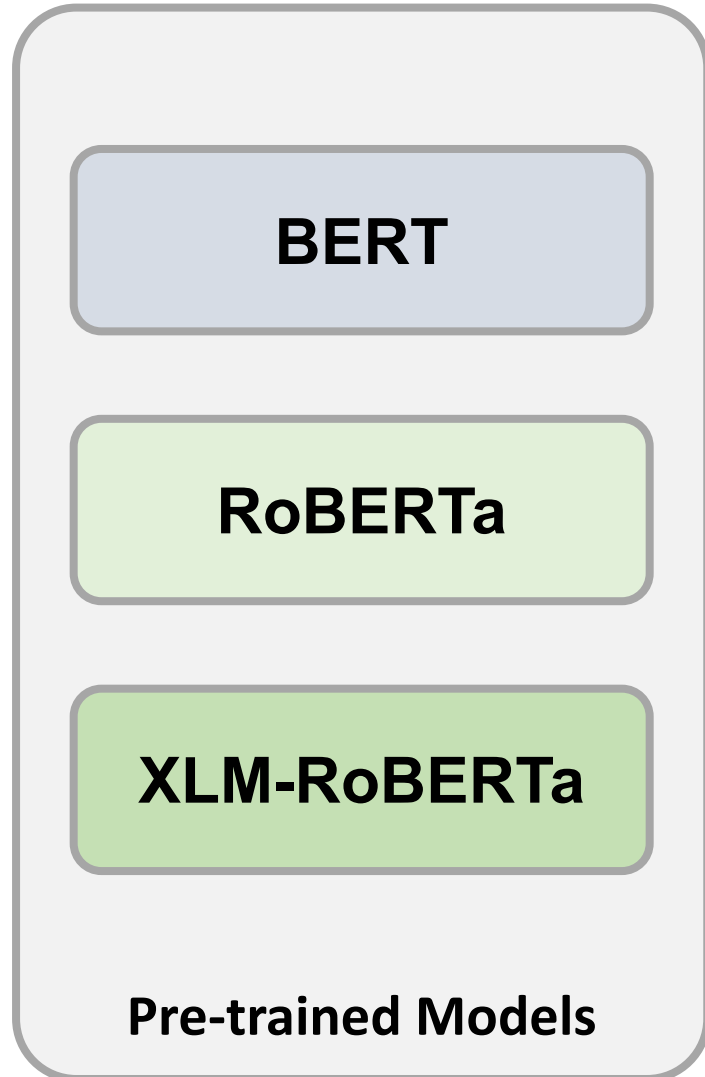
2020 NTCIR-15 Dialogue Evaluation (DialEval-1) Task

Dialogue Quality (DQ) and Nugget Detection (ND)

Chinese Dialogue Quality (S-score) Results (Zeng et al., 2020)

Run	Mean RSNOD	Run	Mean NMD
IMTKU-run2	0.1918	IMTKU-run2	0.1254
IMTKU-run1	0.1964	IMTKU-run0	0.1284
IMTKU-run0	0.1977	IMTKU-run1	0.1290
TUA1-run2	0.2024	TUA1-run2	0.1310
TUA1-run0	0.2053	TUA1-run0	0.1322
NKUST-run1	0.2057	NKUST-run1	0.1363
BL-lstm	0.2088	TUA1-run1	0.1397
WUST-run0	0.2131	BL-popularity	0.1442
RSLNV-run0	0.2141	BL-lstm	0.1455
BL-popularity	0.2288	RSLNV-run0	0.1483
TUA1-run1	0.2302	WUST-run0	0.1540
NKUST-run0	0.2653	NKUST-run0	0.2289
BL-uniform	0.2811	BL-uniform	0.2497

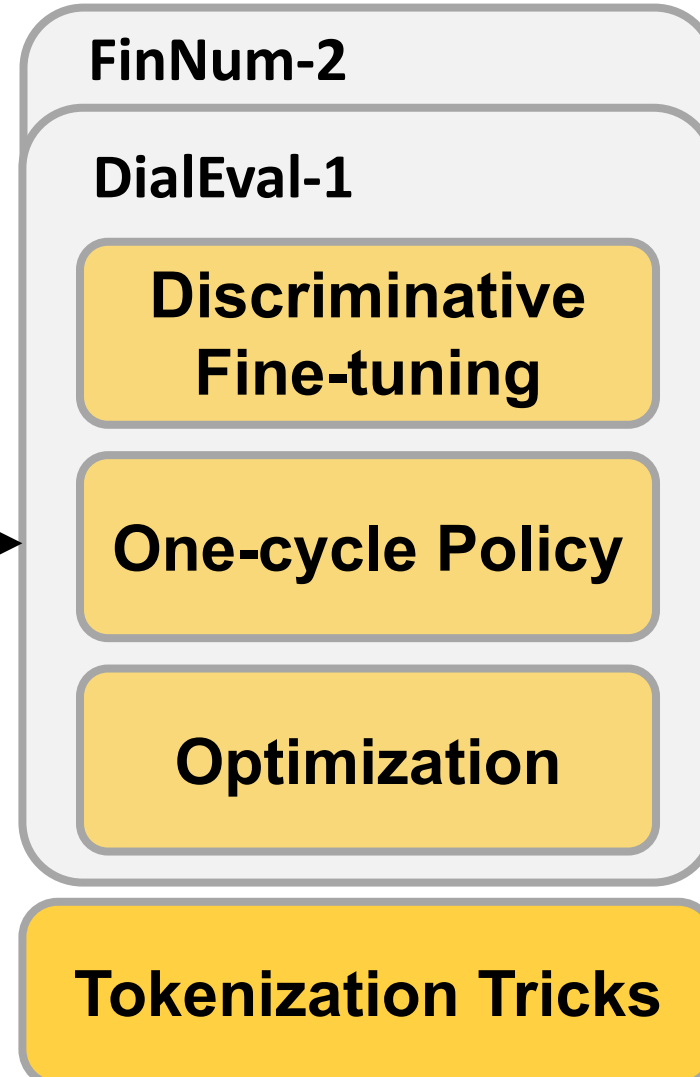
Transformer-based Models Selection



Transfer Learning



Fine-tuning Techniques





**Short Text Conversation Task
(STC-3)
Chinese Emotional
Conversation Generation
(CECG) Subtask**

NTCIR Short Text Conversation

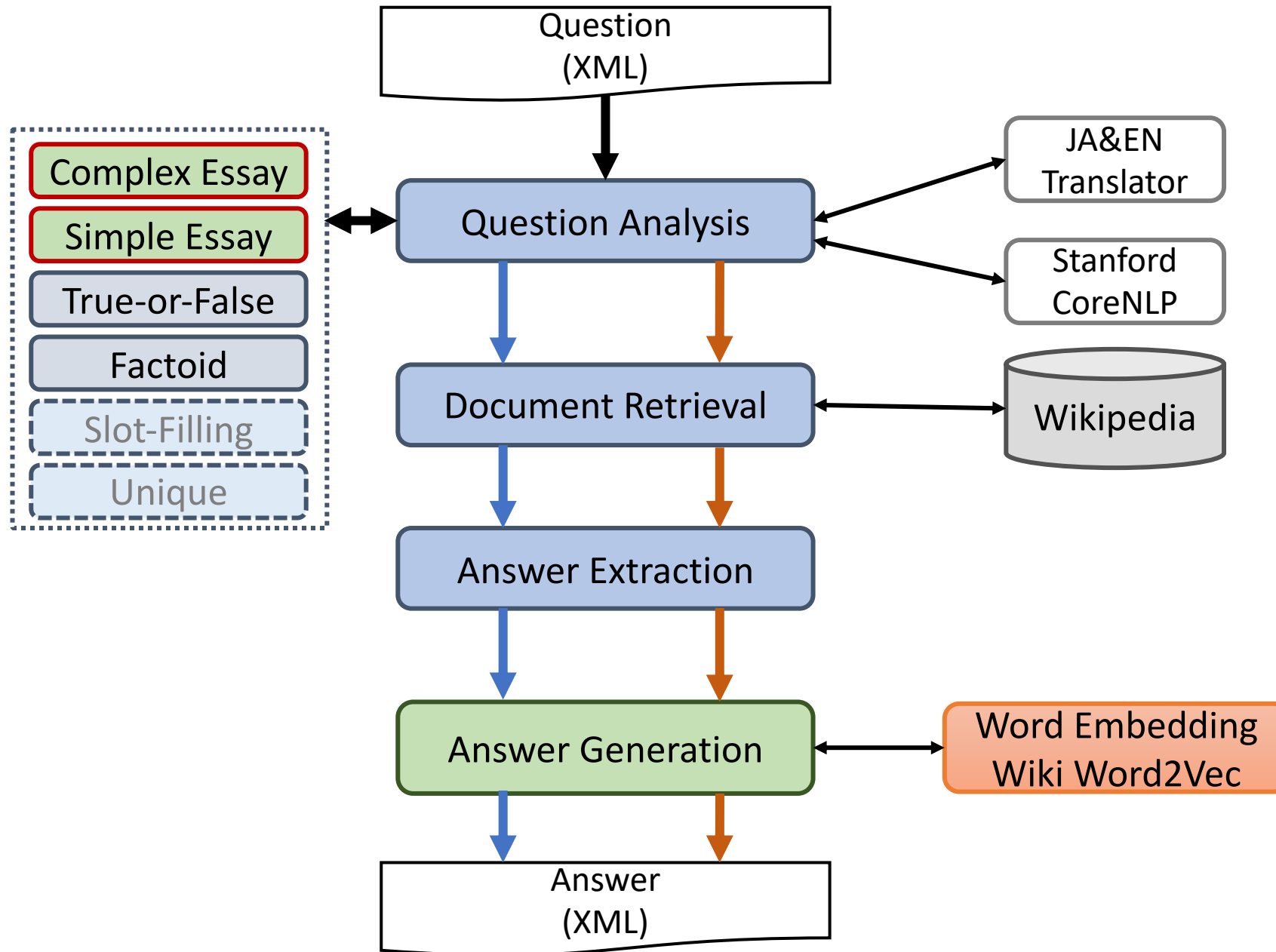
STC-1, STC-2, STC-3

	Japanese	Chinese	English	
NTCIR-12 STC-1 22 active participants	Twitter, Retrieval	Weibo, Retrieval		Single-turn, Non task-oriented
NTCIR-13 STC-2 27 active participants	Yahoo! News, Retrieval+ Generation	Weibo, Retrieval+ Generation		
NTCIR-14 STC-3		Weibo, Generation for given emotion categories		Multi-turn, task-oriented (helpdesk)
		Weibo+English translations, distribution estimation for subjective annotations		

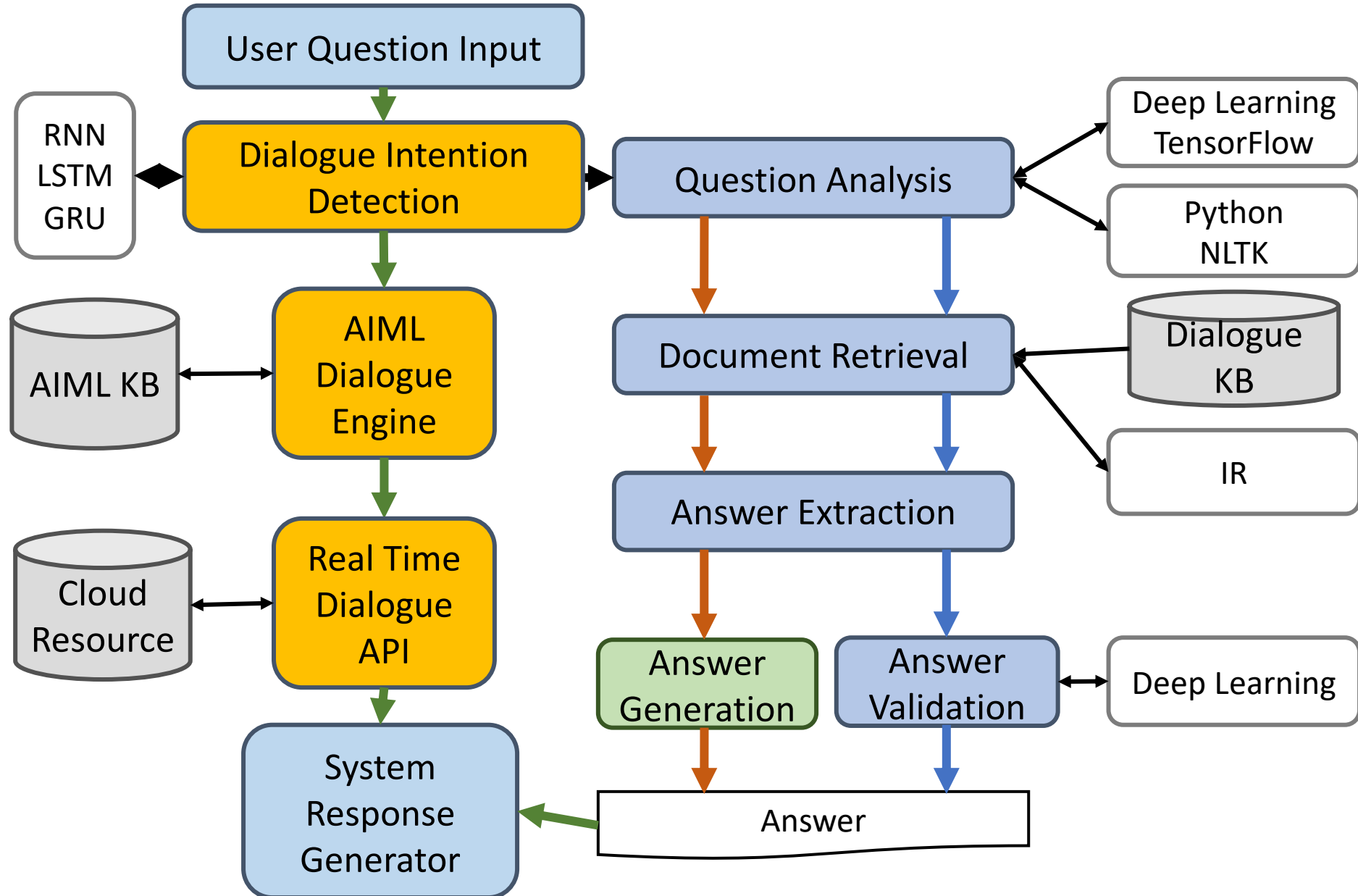
Chinese Emotional Conversation Generation (CECG) subtask

Dialogue Quality (DQ) and Nugget Detection (ND) subtasks

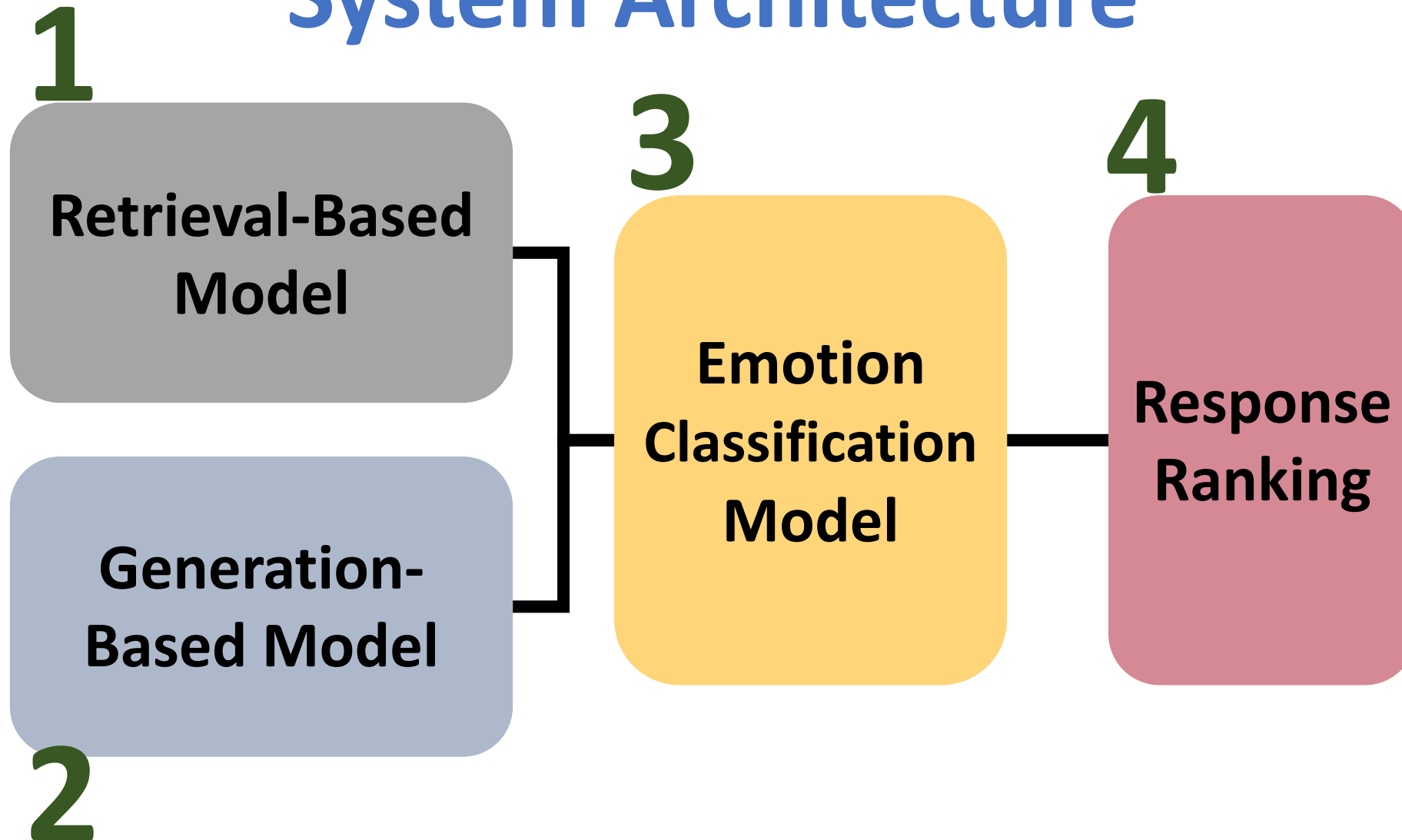
IMTKU System Architecture for NTCIR-13 QALab-3



System Architecture of Intelligent Dialogue and Question Answering System



IMTKU Emotional Dialogue System Architecture

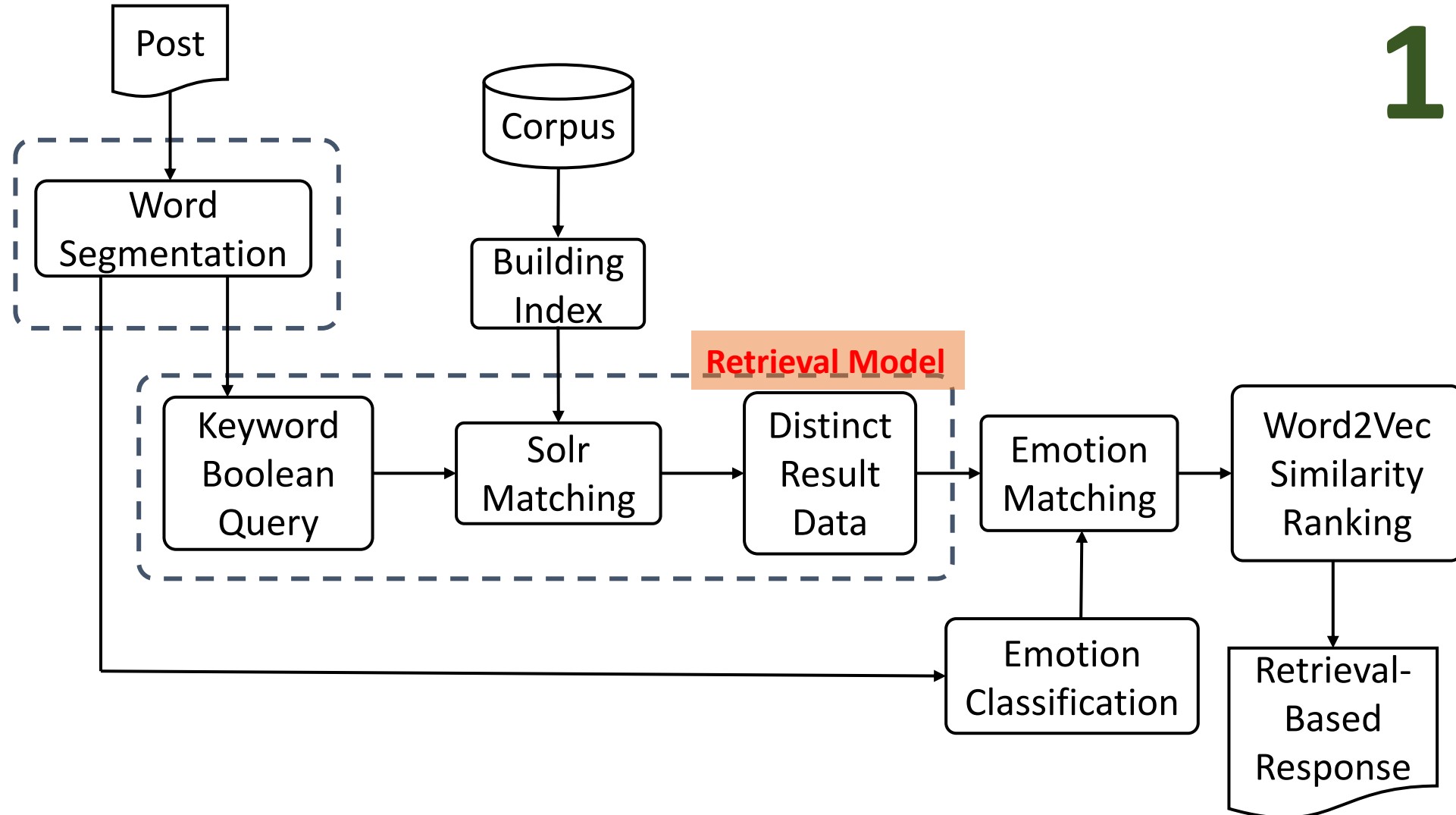




The system architecture of IMTKU retrieval-based model for NTCIR-14 STC-3

Retrieval-Based Model

1

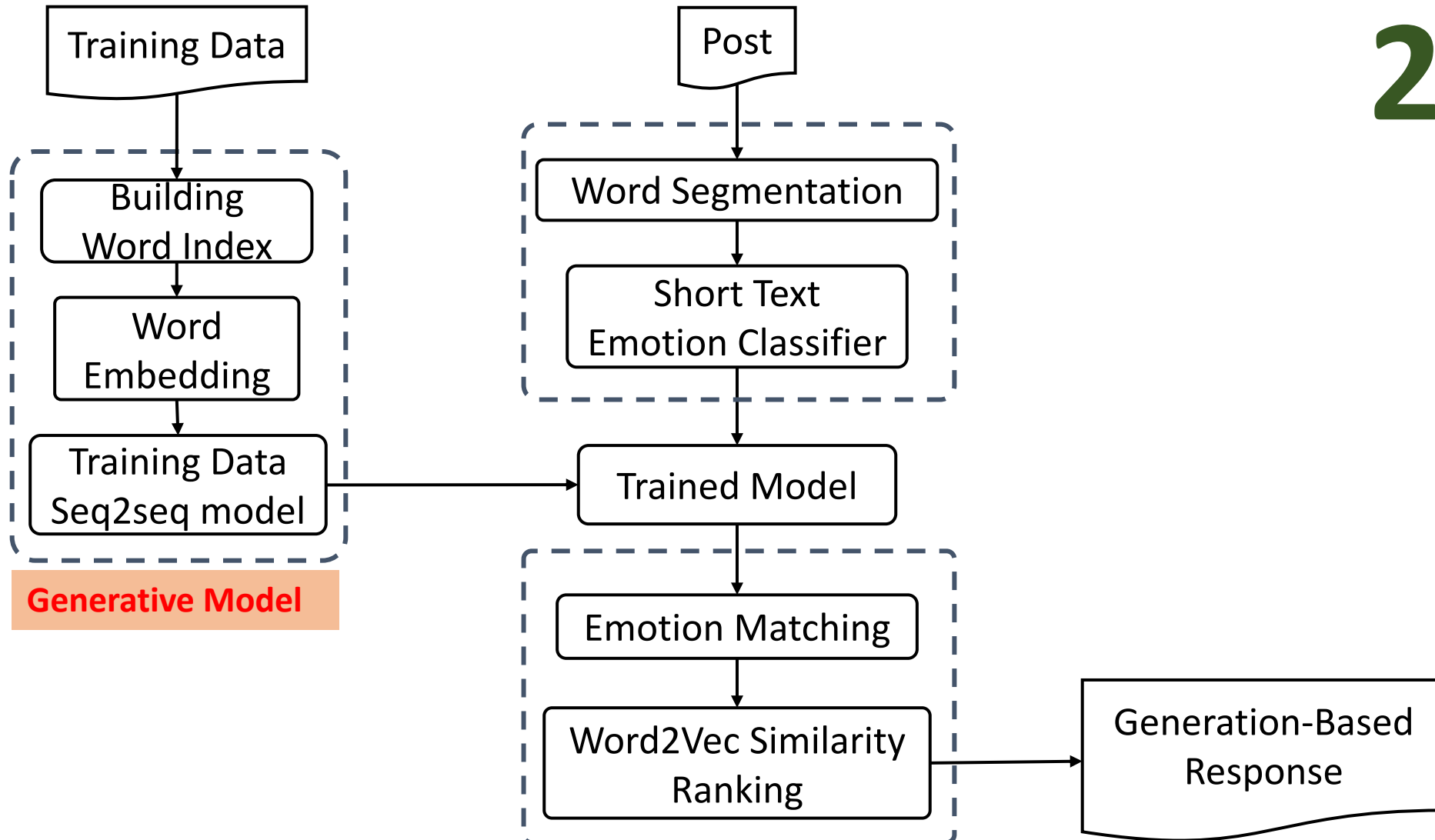


The system architecture of

IMTKU generation-based model for NTCIR-14 STC-3



Generation-Based Model

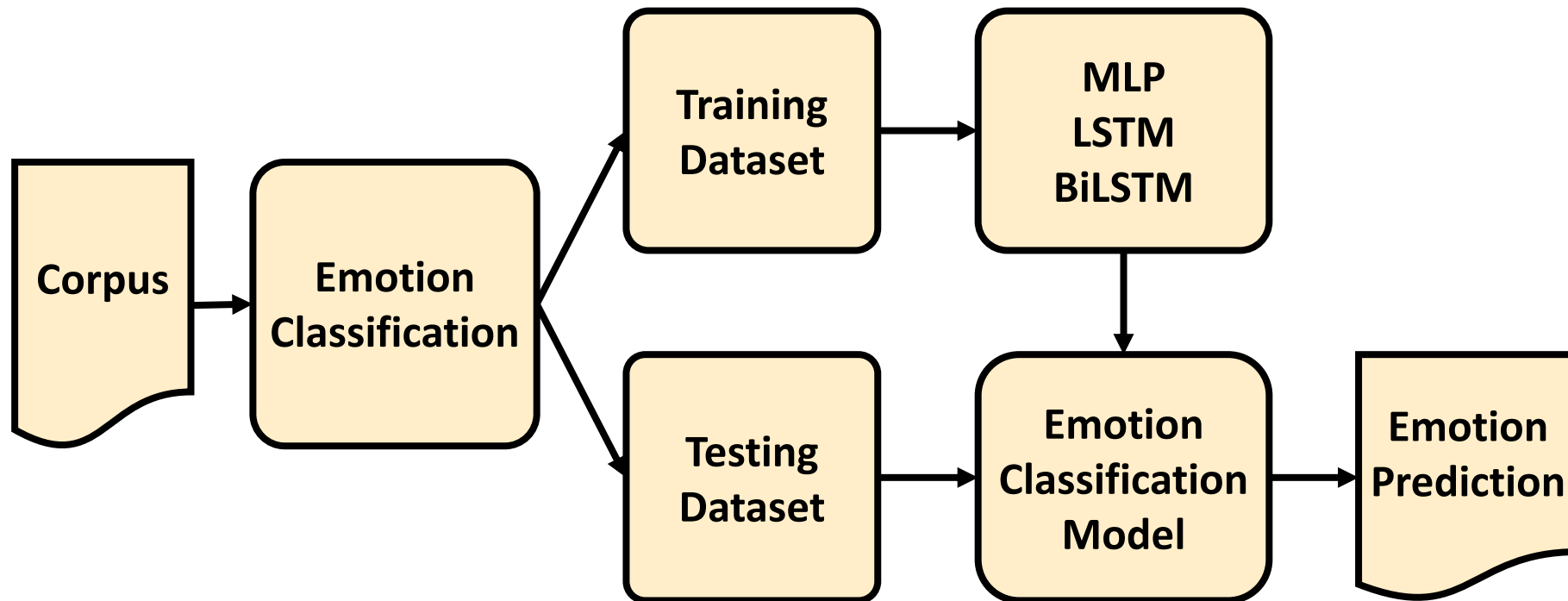


The system architecture of IMTKU emotion classification model for NTCIR-14 STC-3



Emotion Classification Model

3

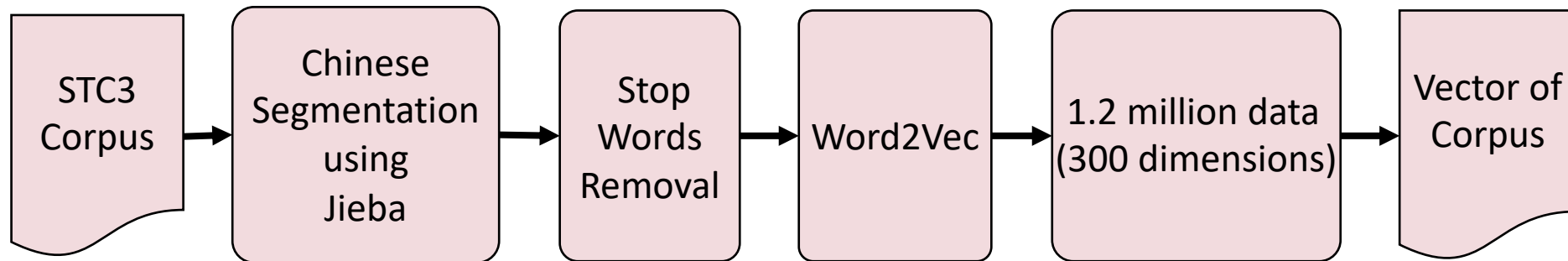


The system architecture of IMTKU Response Ranking for NTCIR-14 STC-3



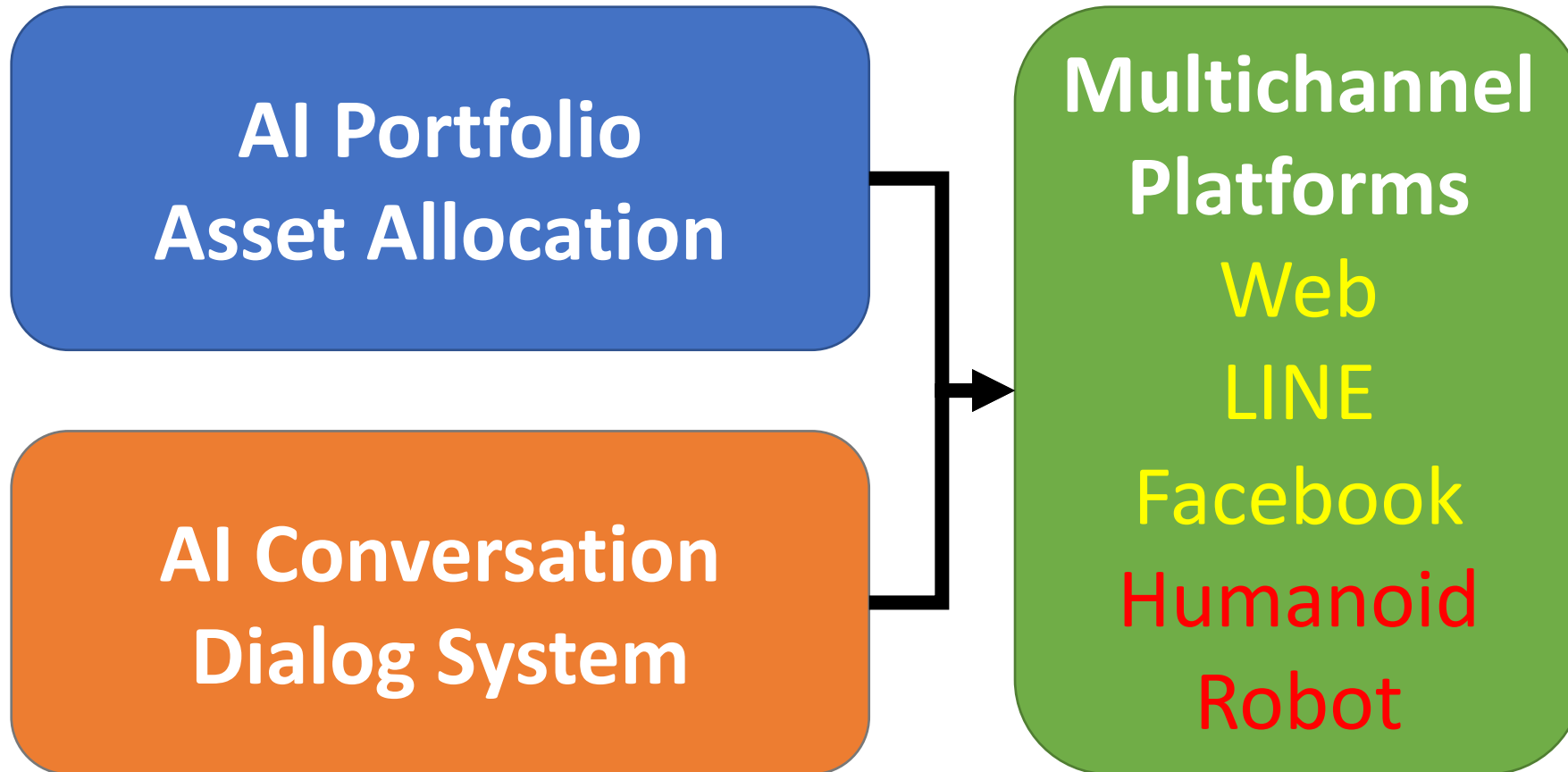
Response Ranking

4

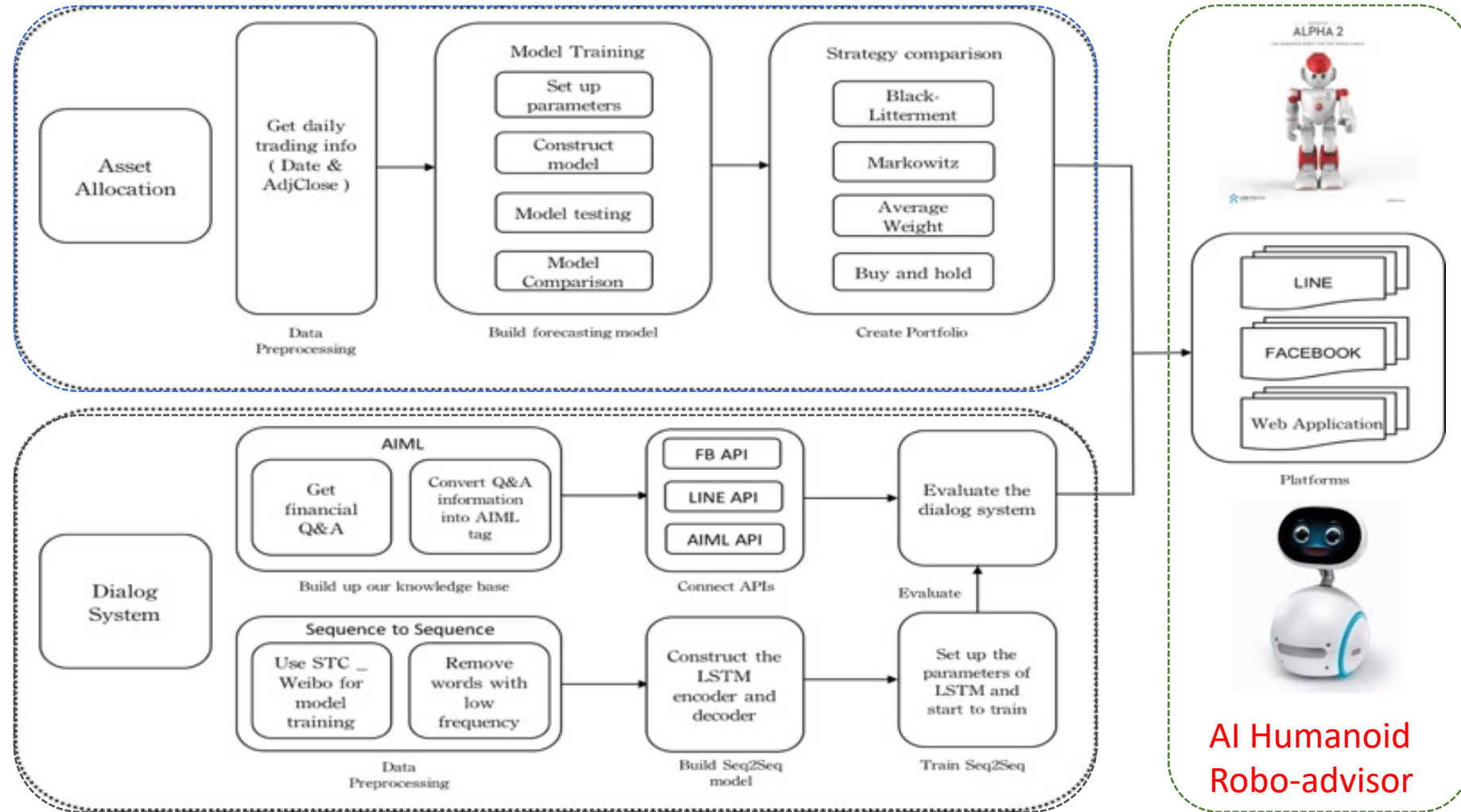


AI Humanoid Robo-Advisor

AI Humanoid Robo-Advisor for Multi-channel Conversational Commerce

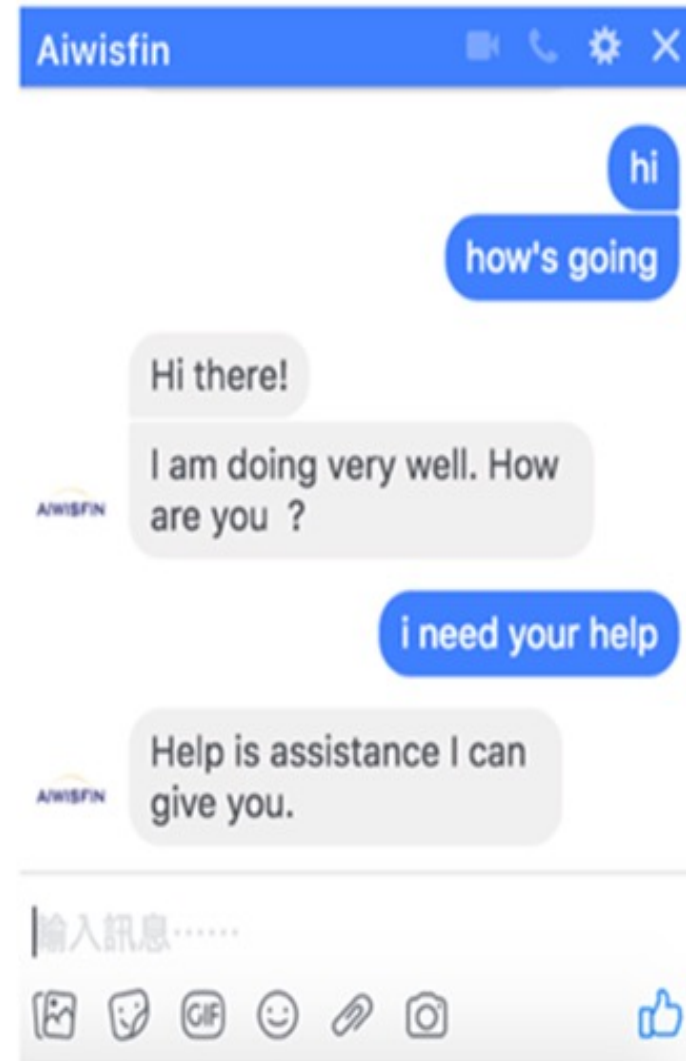


System Architecture of AI Humanoid Robo-Advisor



AI Humanoid Robo-advisor

Conversational Model (LINE, FB Messenger)



Conversational Robo-Advisor Multichannel UI/UX Robots



ALPHA 2



ZENBO

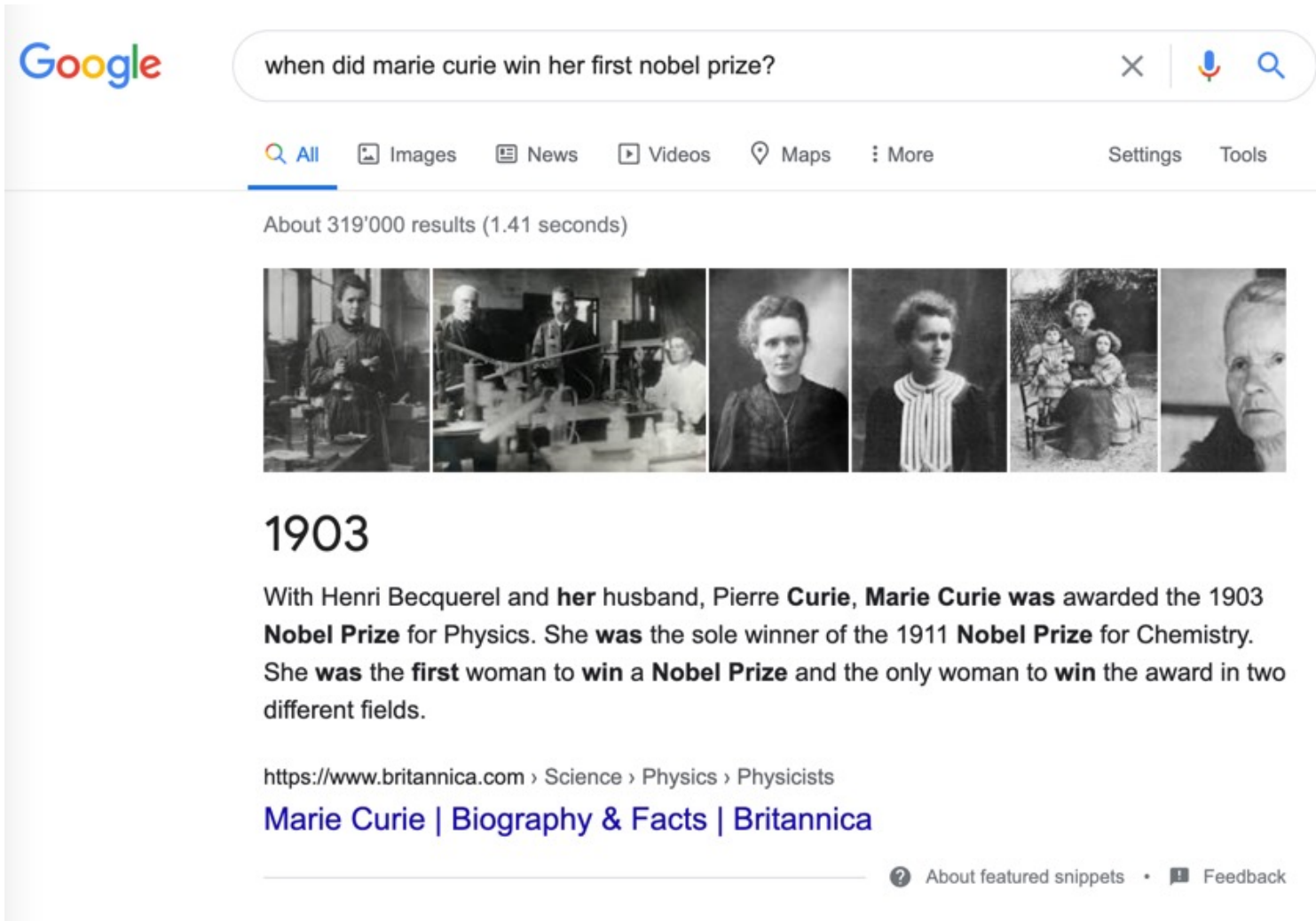


Question Answering

Question Answering

When did Marie Curie win her first Nobel Prize?

1903

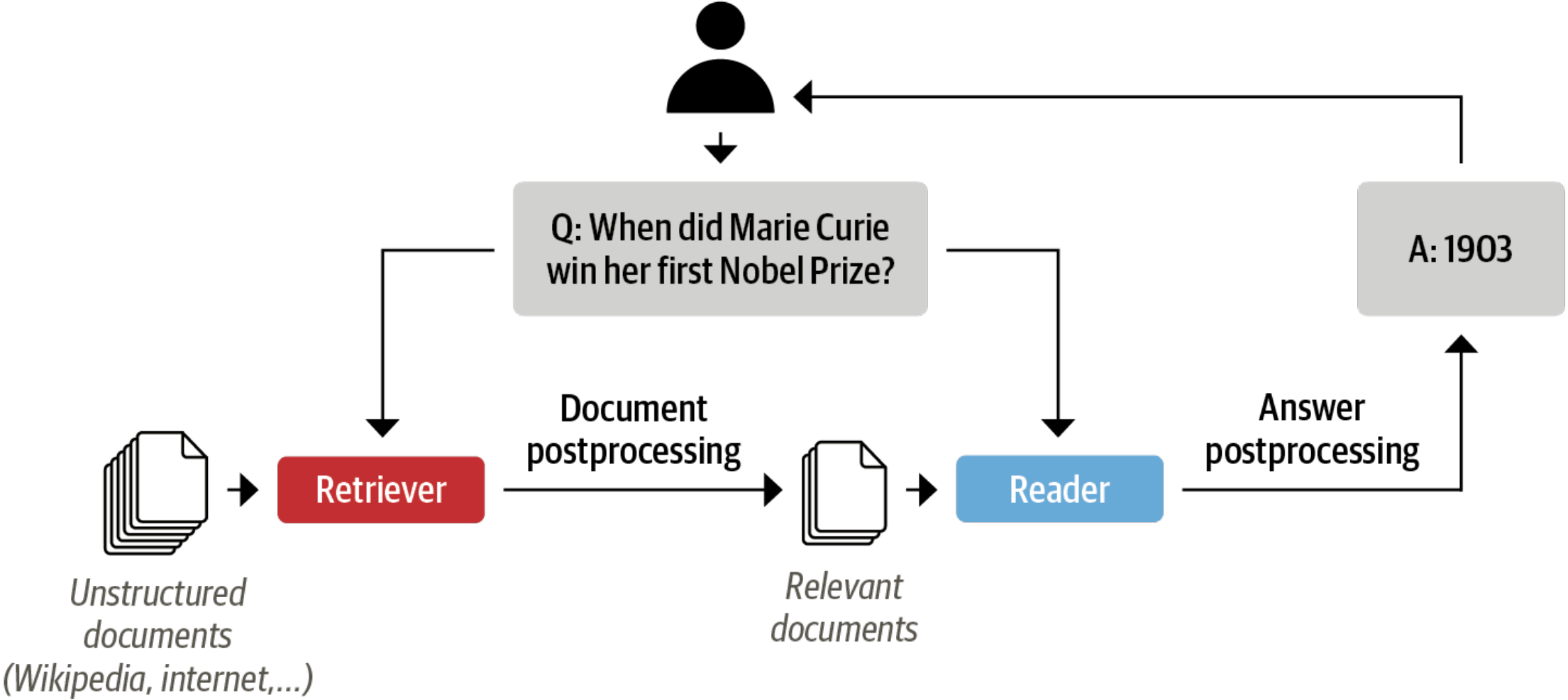


The image shows a Google search interface. The search bar contains the text "when did marie curie win her first nobel prize?". Below the search bar, there are navigation options: "All", "Images", "News", "Videos", "Maps", "More", "Settings", and "Tools". The search results show "About 319'000 results (1.41 seconds)". A row of six small images is displayed, showing various portraits and scenes related to Marie Curie. Below the images, the year "1903" is prominently displayed. A text snippet follows: "With Henri Becquerel and her husband, Pierre Curie, Marie Curie was awarded the 1903 Nobel Prize for Physics. She was the sole winner of the 1911 Nobel Prize for Chemistry. She was the first woman to win a Nobel Prize and the only woman to win the award in two different fields." Below this snippet is a link to a Britannica article: "https://www.britannica.com › Science › Physics › Physicists Marie Curie | Biography & Facts | Britannica". At the bottom right of the search results, there are links for "About featured snippets" and "Feedback".

The Retriever-Reader Architecture for Modern QA Systems

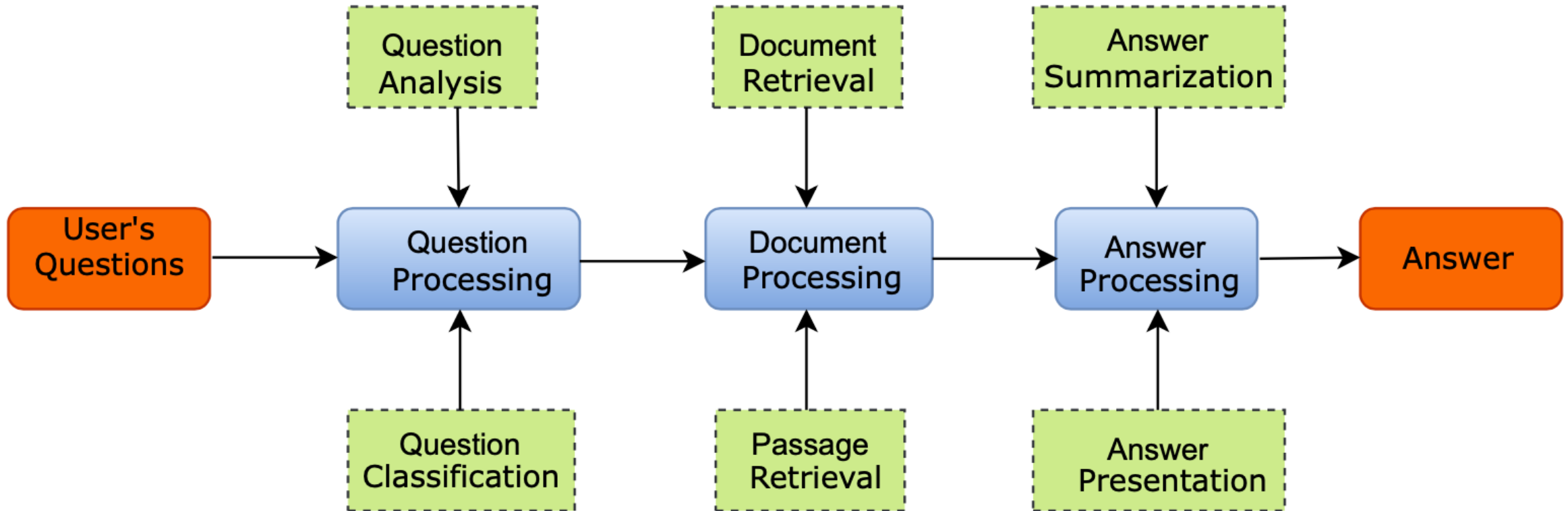
When did Marie Curie win her first Nobel Prize?

1903

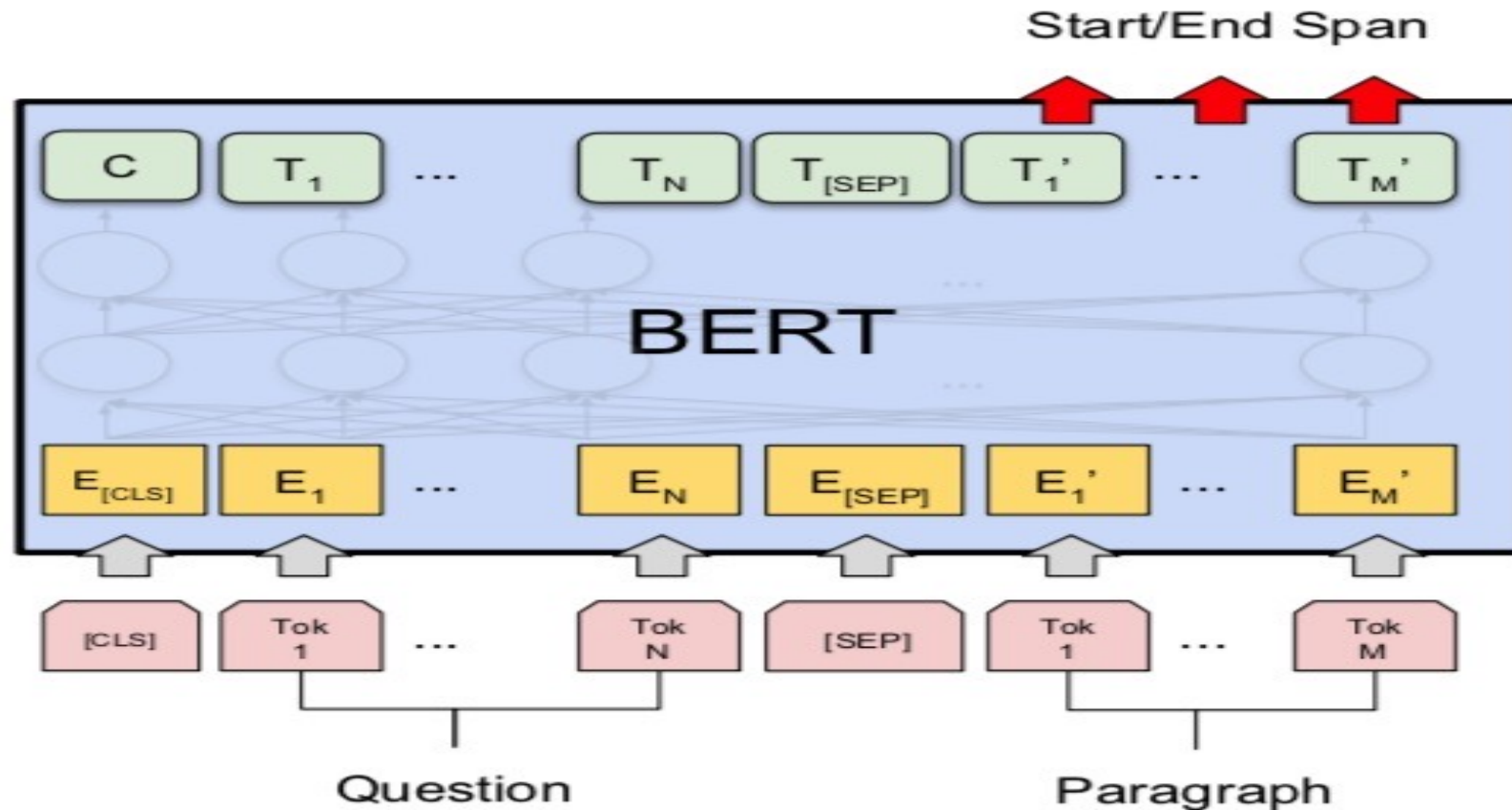


Source: Lewis Tunstall, Leandro von Werra, and Thomas Wolf (2022), Natural Language Processing with Transformers: Building Language Applications with Hugging Face, O'Reilly Media.

Question Answering System (QAS)

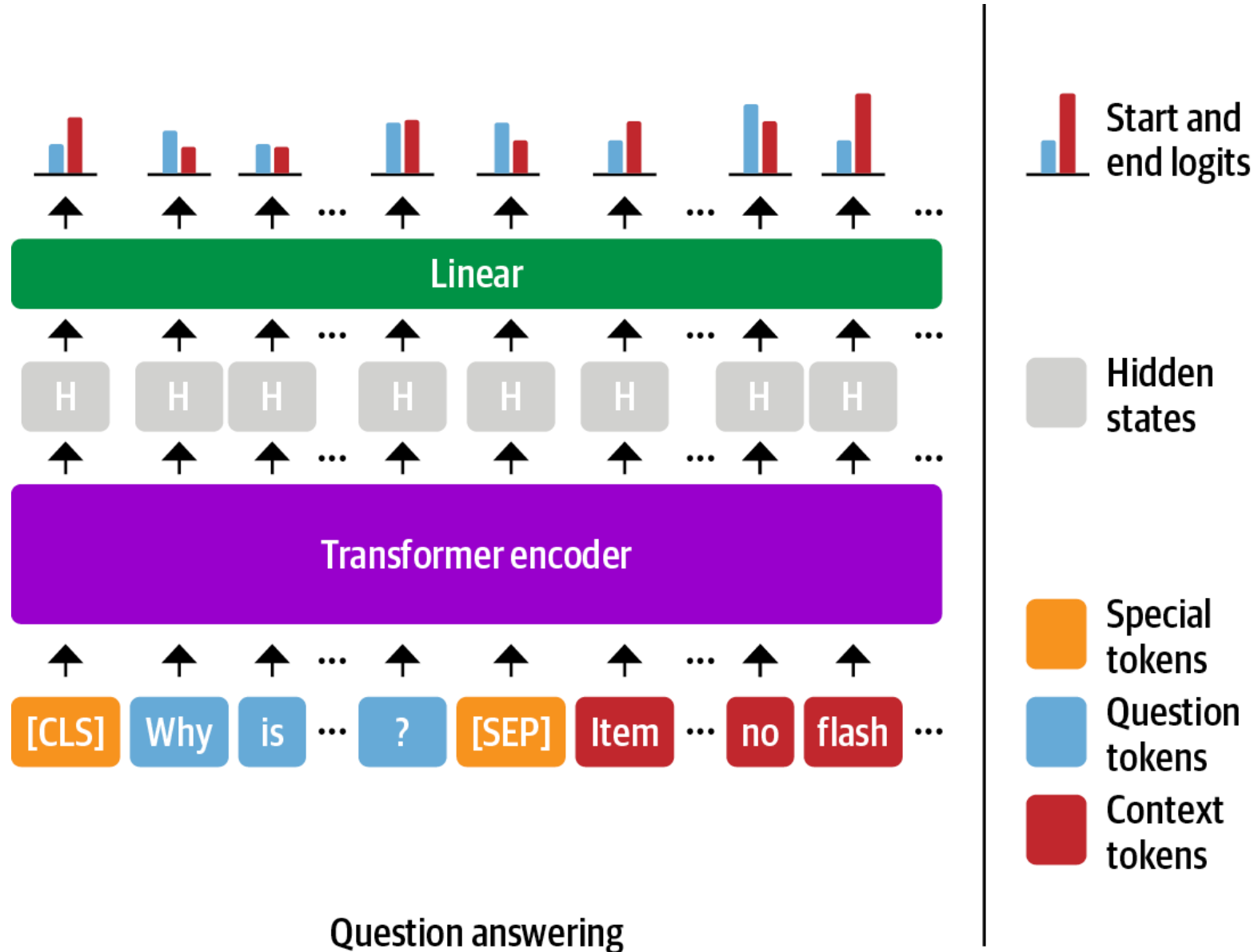


Fine-tuning BERT on Question Answering (QA)



(c) Question Answering Tasks:
SQuAD v1.1

The span classification head for QA tasks

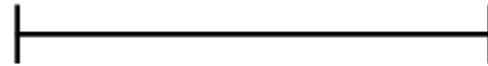


Multiple question-context pairs

Why is the camera of poor quality? Item like the picture, fast deliver 3 days well packed, good quality for the price. The camera is decent (as phone cameras go). There is no flash though...



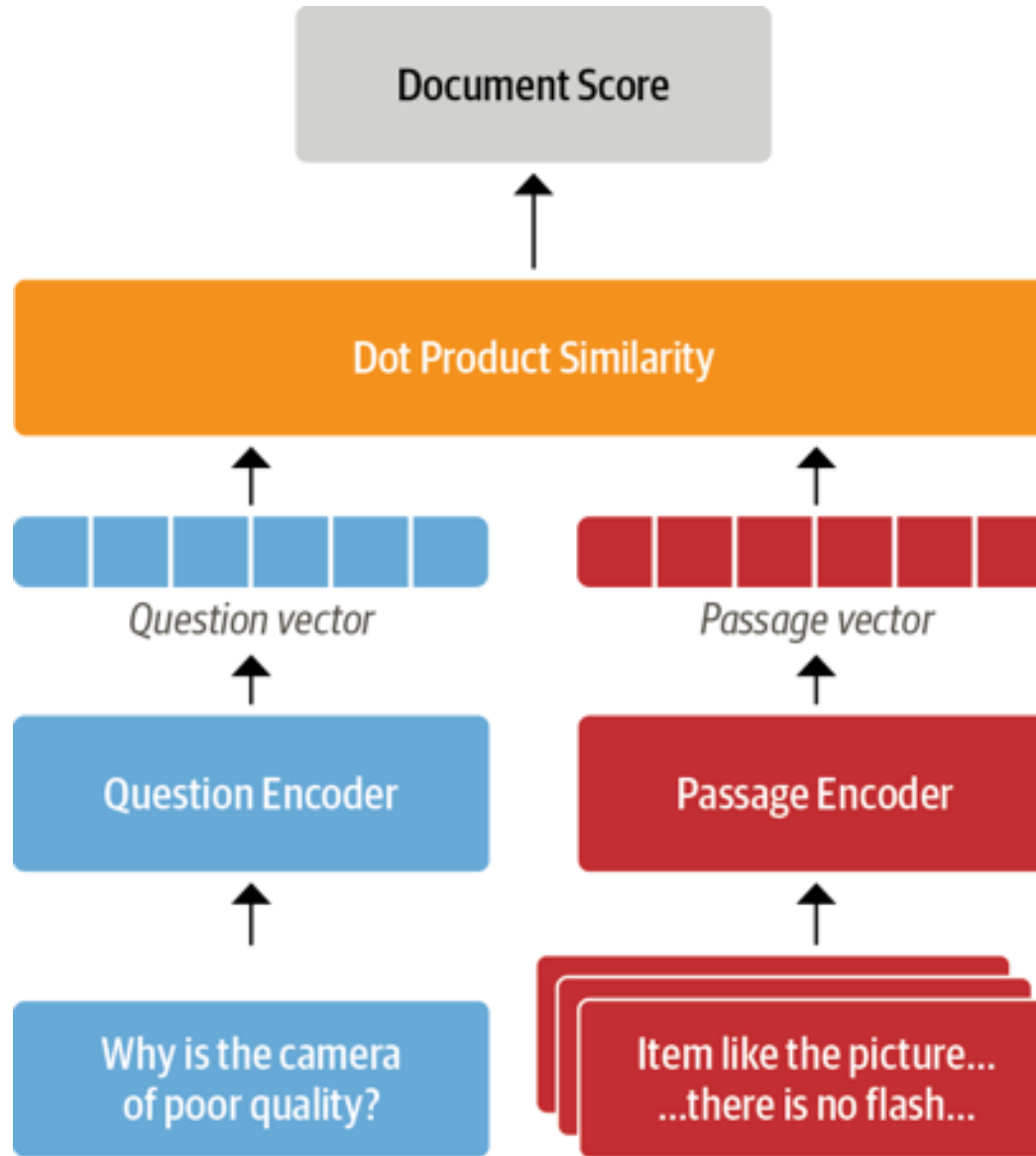
Stride



Why is the camera of poor quality? Item like the picture, fast deliver 3 days well packed, good quality for the price. The camera is decent (as phone cameras go). There is no flash though...



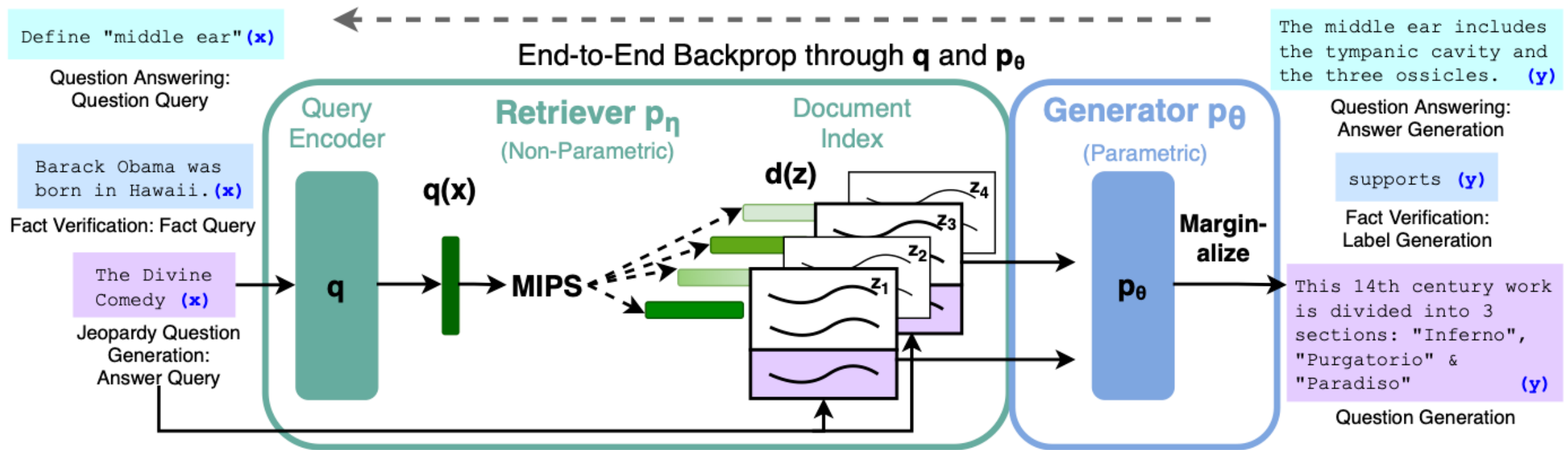
Dense Passage Retrieval (DPR)



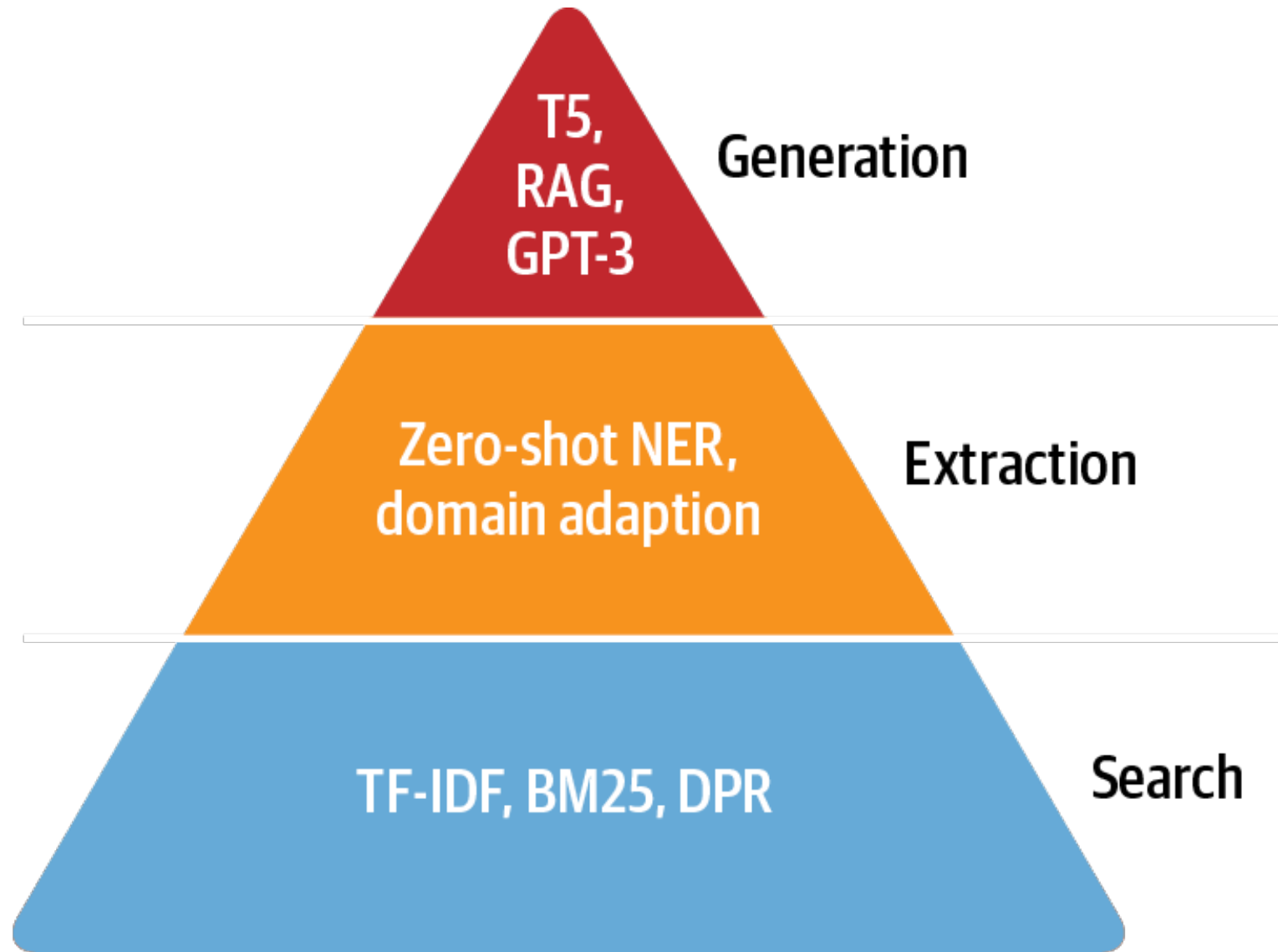
Going Beyond Extractive QA

Retrieval-Augmented Generation (RAG)

The RAG architecture for fine-tuning a retriever and generator end-to-end

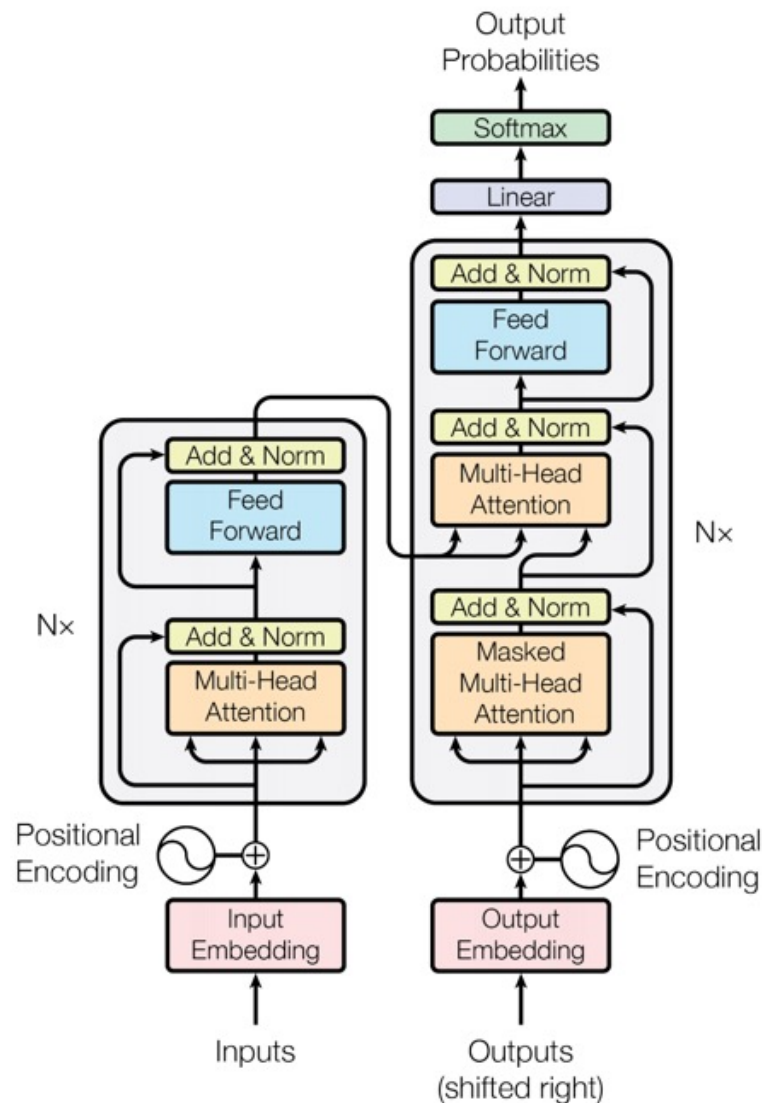


The QA Hierarchy of Needs



Transformer (Attention is All You Need)

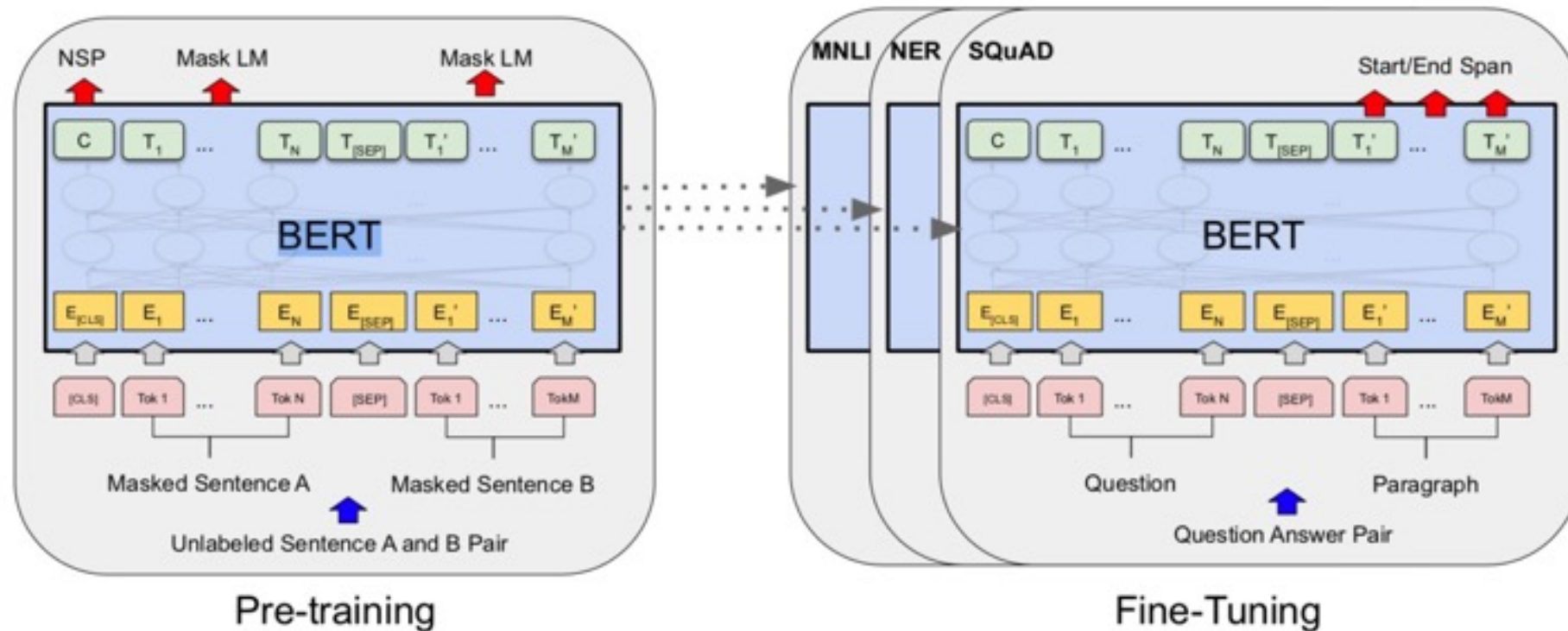
(Vaswani et al., 2017)



BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

BERT (Bidirectional Encoder Representations from Transformers)

Overall pre-training and fine-tuning procedures for BERT



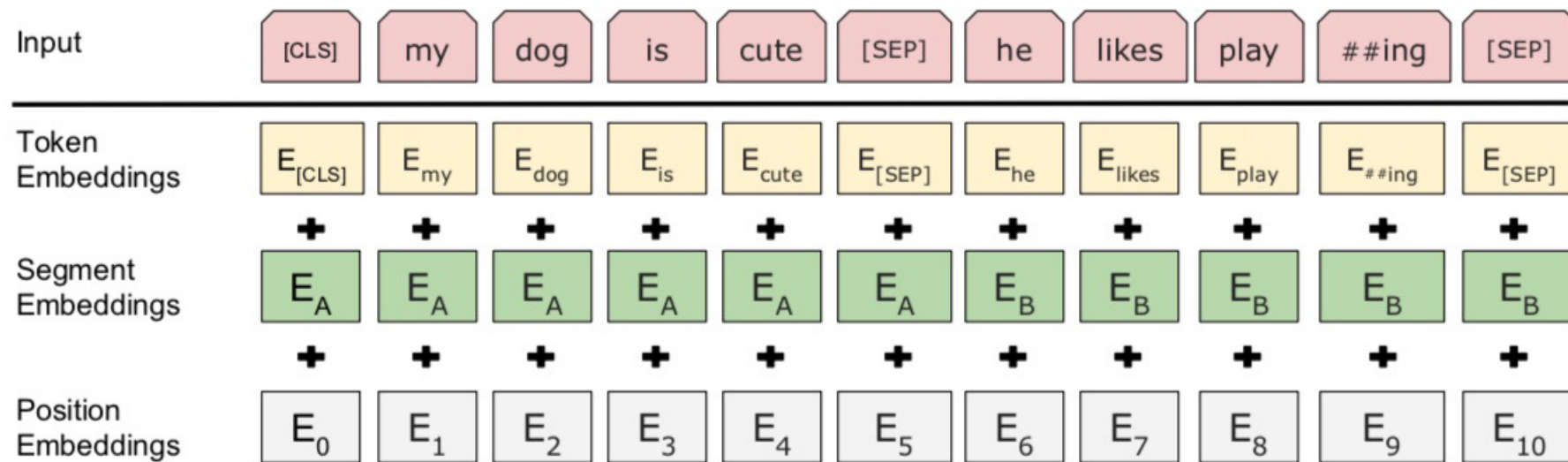
Source: Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova (2018).

"Bert: Pre-training of deep bidirectional transformers for language understanding." arXiv preprint arXiv:1810.04805.

BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

BERT (Bidirectional Encoder Representations from Transformers)

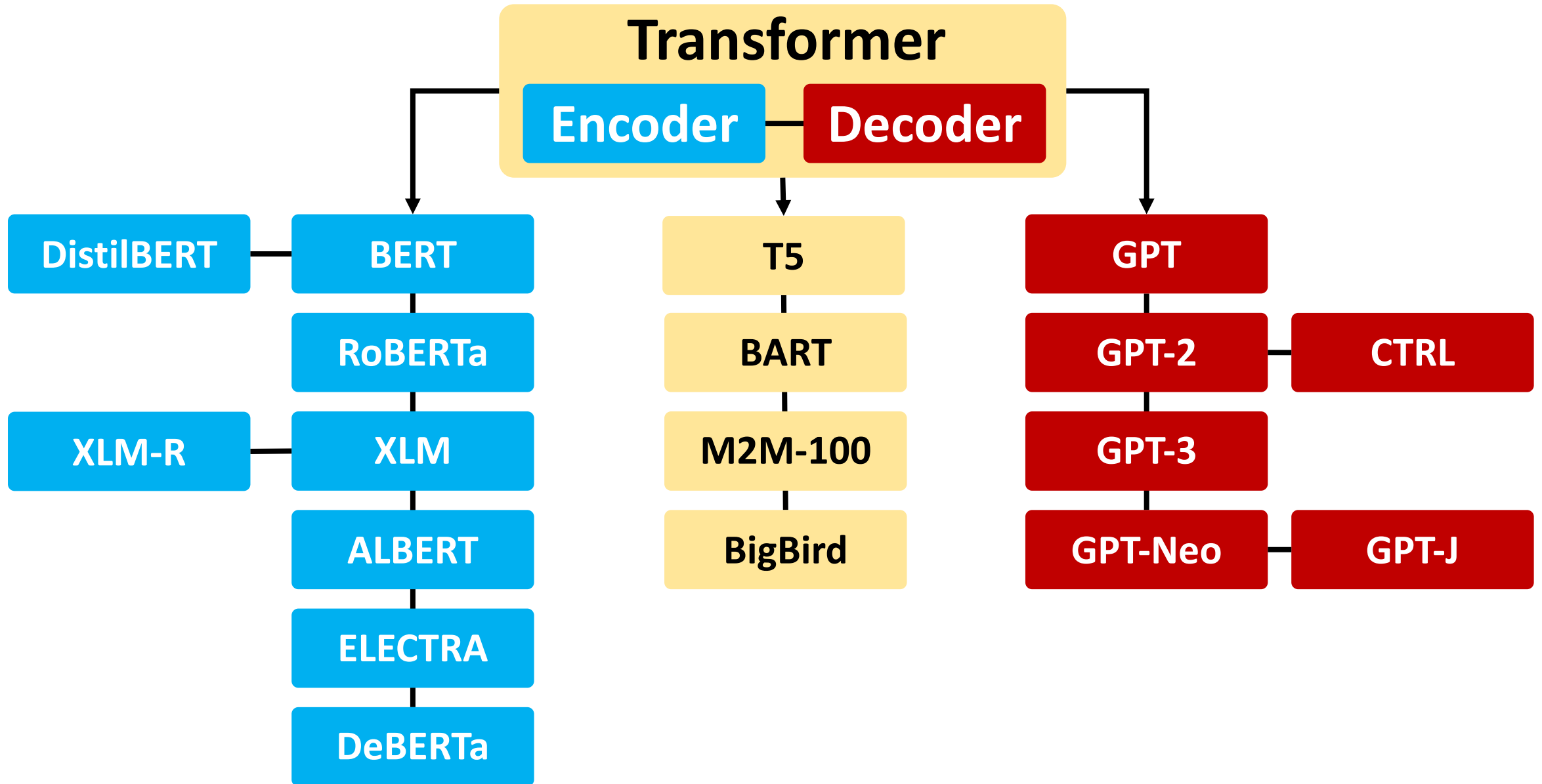
BERT input representation



Source: Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova (2018).

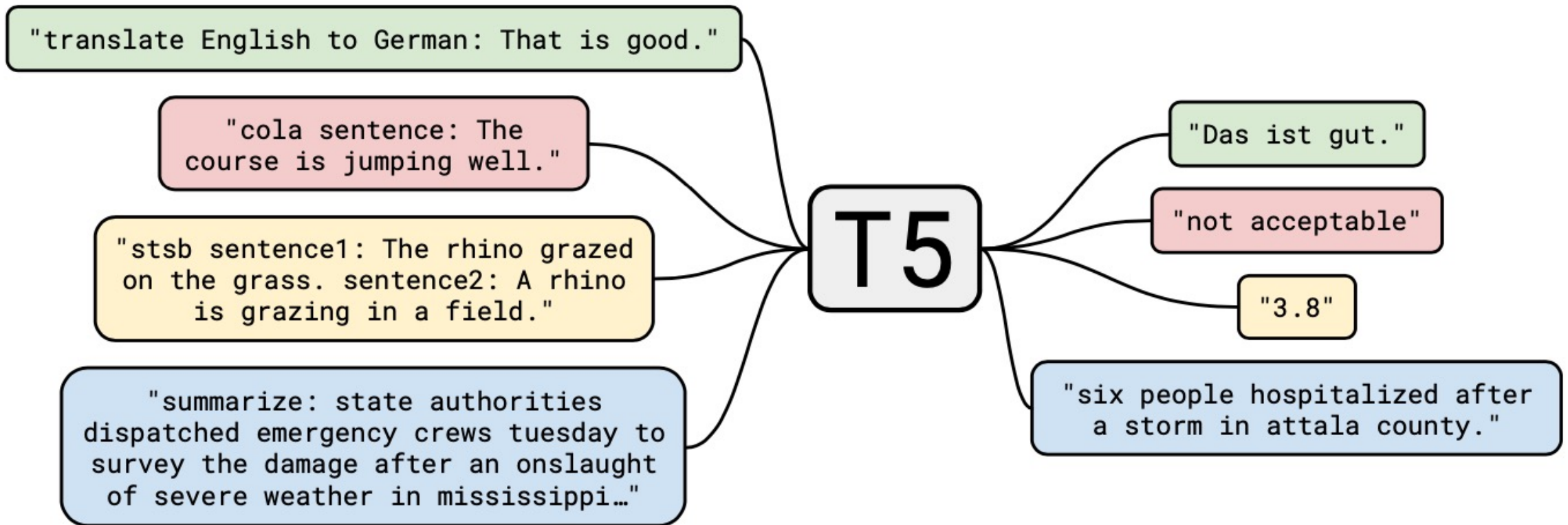
"Bert: Pre-training of deep bidirectional transformers for language understanding." arXiv preprint arXiv:1810.04805.

Transformer Models

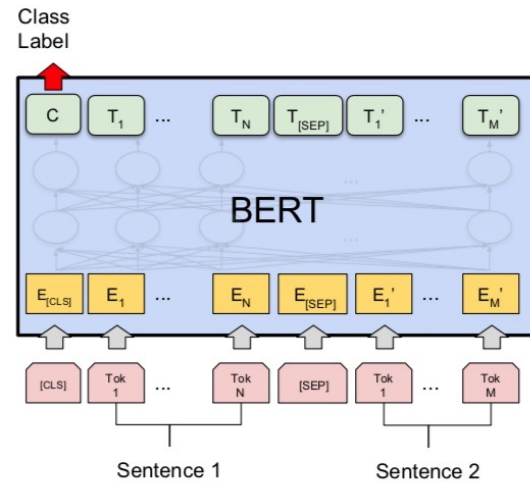


T5

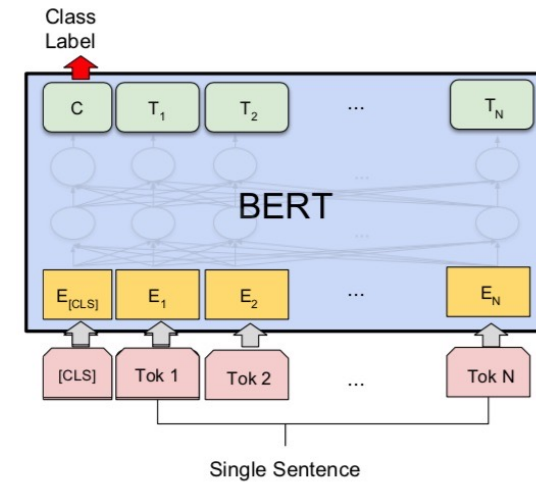
Text-to-Text Transfer Transformer



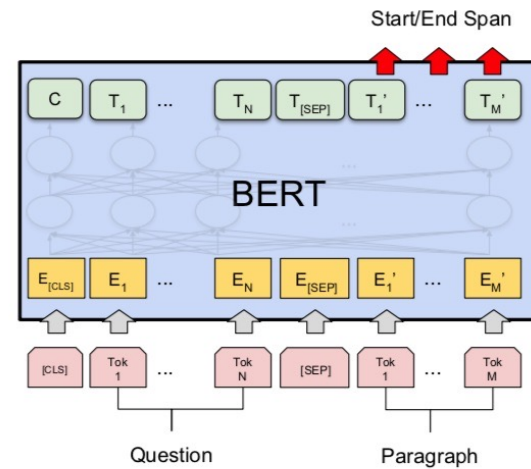
Fine-tuning BERT on Different Tasks



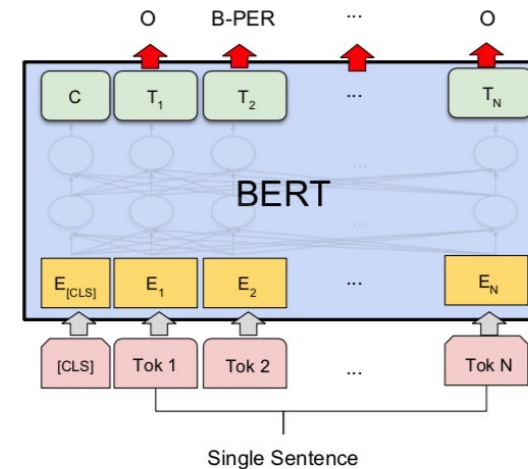
(a) Sentence Pair Classification Tasks:
MNLI, QQP, QNLI, STS-B, MRPC,
RTE, SWAG



(b) Single Sentence Classification Tasks:
SST-2, CoLA



(c) Question Answering Tasks:
SQuAD v1.1

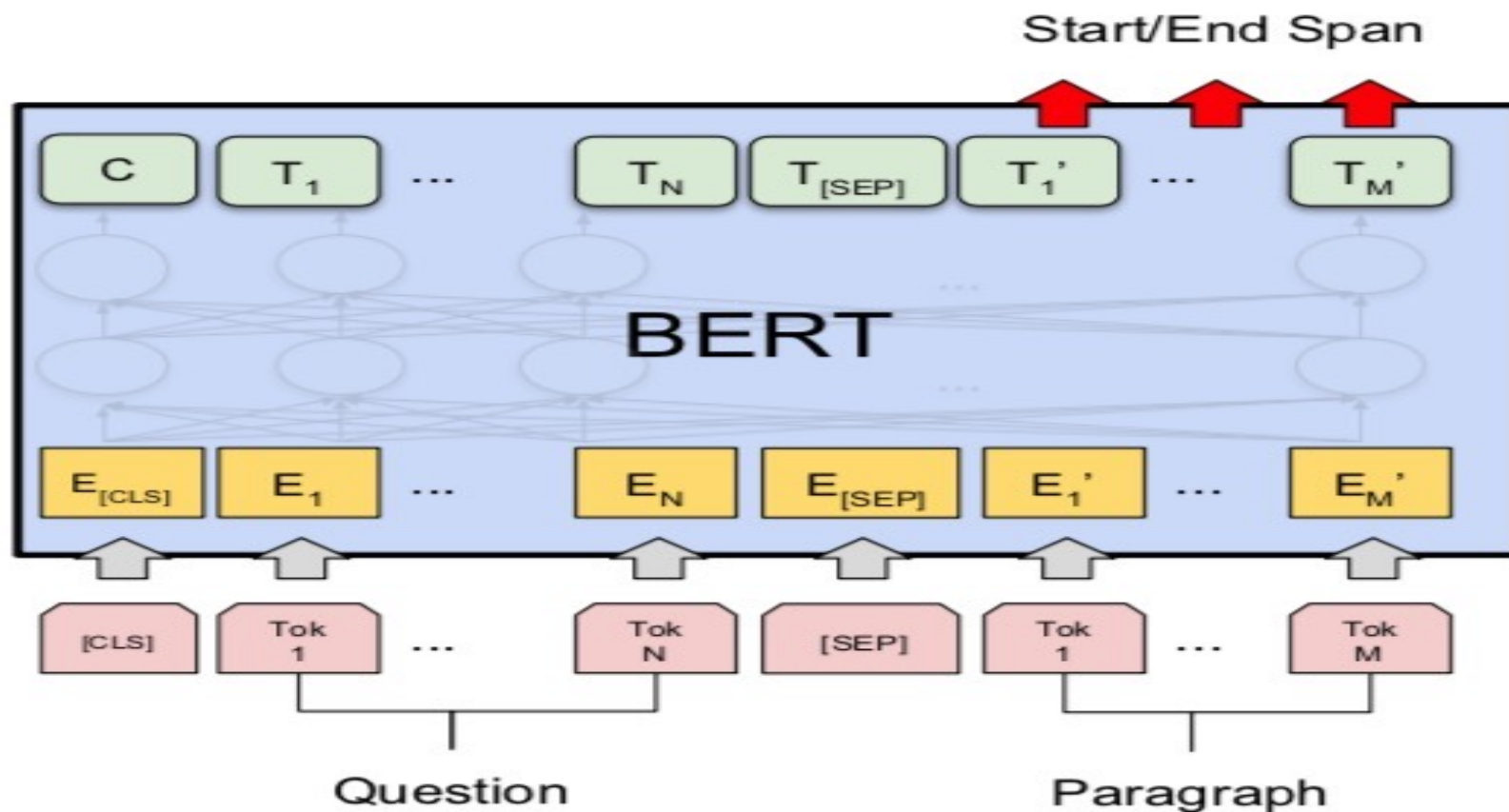


(d) Single Sentence Tagging Tasks:
CoNLL-2003 NER

Source: Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova (2018).

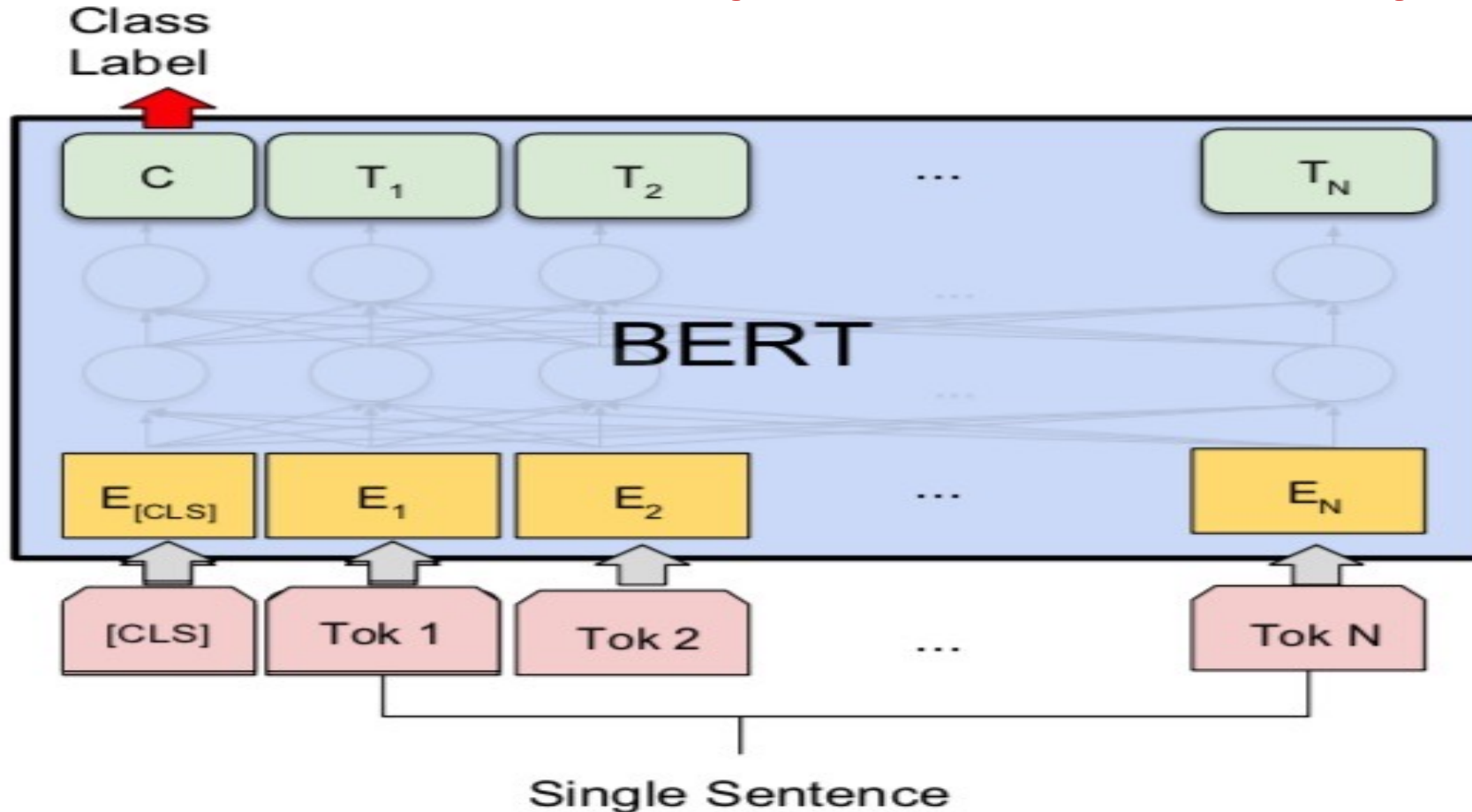
"Bert: Pre-training of deep bidirectional transformers for language understanding." arXiv preprint arXiv:1810.04805.

Fine-tuning BERT on Question Answering (QA)



(c) Question Answering Tasks:
SQuAD v1.1

Fine-tuning BERT on Dialogue Intent Detection (ID; Classification)



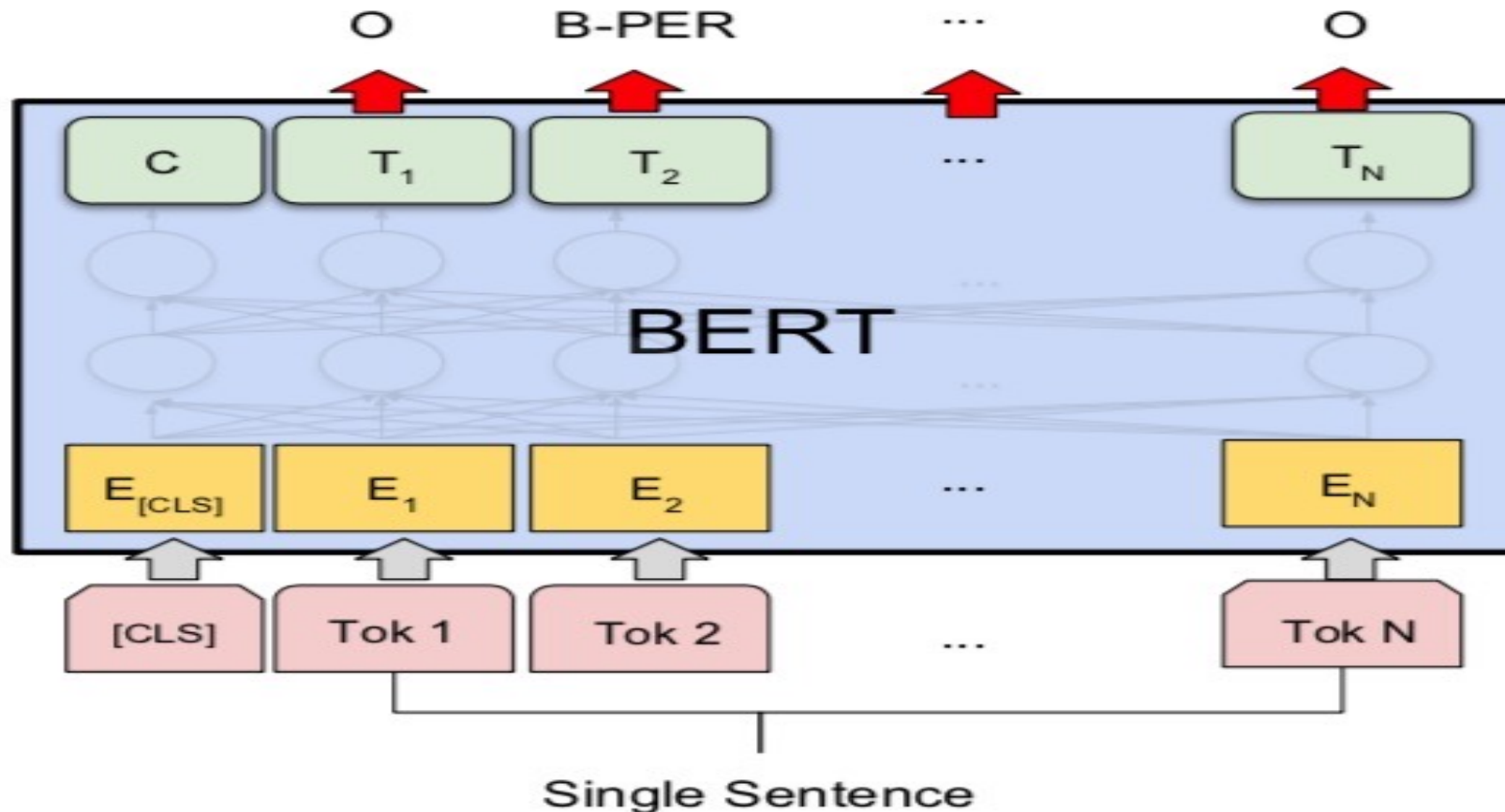
(b) Single Sentence Classification Tasks: SST-2, CoLA

Source: Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova (2018).

"Bert: Pre-training of deep bidirectional transformers for language understanding." arXiv preprint arXiv:1810.04805.

Fine-tuning BERT on Dialogue

Slot Filling (SF)



(d) Single Sentence Tagging Tasks:
CoNLL-2003 NER

Source: Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova (2018).

"Bert: Pre-training of deep bidirectional transformers for language understanding." arXiv preprint arXiv:1810.04805.

Question Answering

(QA)

SQuAD

Stanford Question Answering Dataset

SQuAD

SQuAD

Home

Explore 2.0

Explore 1.1

SQuAD2.0

The Stanford Question Answering Dataset

What is SQuAD?

Stanford **Q**uestion **A**nswering **D**ataset (SQuAD) is a reading comprehension dataset, consisting of questions posed by crowdworkers on a set of Wikipedia articles, where the answer to every question is a segment of text, or *span*, from the corresponding reading passage, or the question might be unanswerable.

SQuAD2.0 combines the 100,000 questions in SQuAD1.1 with over 50,000 unanswerable questions written adversarially by crowdworkers to look similar to answerable ones. To do well on SQuAD2.0, systems must not only answer questions when possible, but also determine when no answer is supported by the paragraph and abstain from answering.

Leaderboard

SQuAD2.0 tests the ability of a system to not only answer reading comprehension questions, but also abstain when presented with a question that cannot be answered based on the provided paragraph.

Rank	Model	EM	F1
	Human Performance Stanford University (Rajpurkar & Jia et al. '18)	86.831	89.452
1 Apr 06, 2020	SA-Net on Albert (ensemble) QIANXIN	90.724	93.011
2 May 05, 2020	SA-Net-V2 (ensemble) QIANXIN	90.679	92.948
?	Retro-Reader (ensemble)	90.578	92.978

<https://rajpurkar.github.io/SQuAD-explorer/>

SQuAD

SQuAD: 100,000+ Questions for Machine Comprehension of Text

Pranav Rajpurkar and Jian Zhang and Konstantin Lopyrev and Percy Liang

{pranavs, zjian, klopyrev, pliang}@cs.stanford.edu

Computer Science Department

Stanford University

Abstract

We present the Stanford Question Answering Dataset (SQuAD), a new reading comprehension dataset consisting of 100,000+ questions posed by crowdworkers on a set of Wikipedia articles, where the answer to each question is a segment of text from the corresponding reading passage. We analyze the dataset to understand the types of reasoning required to answer the questions, leaning heavily on dependency and constituency trees. We build a strong logistic regression model, which achieves an F1 score of 51.0%, a significant improvement over a simple baseline (20%). However, human performance (86.8%) is much higher, indicating that the dataset presents a good challenge problem for future research. The dataset is freely available at <https://stanford-qa.com>.

In meteorology, precipitation is any product of the condensation of atmospheric water vapor that falls under **gravity**. The main forms of precipitation include drizzle, rain, sleet, snow, **graupel** and hail... Precipitation forms as smaller droplets coalesce via collision with other rain drops or ice crystals **within a cloud**. Short, intense periods of rain in scattered locations are called "showers".

What causes precipitation to fall?

gravity

What is another main form of precipitation besides drizzle, rain, snow, sleet and hail?

graupel

Where do water droplets collide with ice crystals to form precipitation?

within a cloud

Figure 1: Question-answer pairs for a sample passage in the

Source: Rajpurkar, Pranav, Jian Zhang, Konstantin Lopyrev, and Percy Liang.

"Squad: 100,000+ questions for machine comprehension of text." arXiv preprint arXiv:1606.05250 (2016).

SQuAD (Question Answering)

Q: What causes precipitation to fall?

Precipitation

From Wikipedia, the free encyclopedia

For other uses, see [Precipitation \(disambiguation\)](#).

In meteorology, **precipitation** is any product of the condensation of atmospheric water vapor that falls under gravity from clouds.^[2] The main forms of precipitation include drizzle, rain, sleet, snow, ice pellets, graupel and hail. Precipitation occurs when a portion of the atmosphere becomes saturated with water vapor (reaching 100% **relative humidity**), so that the water condenses and "precipitates". Thus, fog and mist are not precipitation but suspensions, because the water vapor does not condense sufficiently to precipitate. Two processes, possibly acting together, can lead to air becoming saturated: cooling the air or adding water vapor to the air. Precipitation forms as smaller droplets coalesce via collision with other rain drops or ice crystals within a cloud. **Short, intense periods of rain in scattered locations are called "showers."**^[3]

SQuAD (Question Answering)

Paragraph

In meteorology, precipitation is any product of the condensation of atmospheric water vapor that falls under gravity. The main forms of precipitation include drizzle, rain, sleet, snow, graupel and hail... Precipitation forms as smaller droplets coalesce via collision with other rain drops or ice crystals within a cloud. Short, intense periods of rain in scattered locations are called “showers”.

Q: What causes precipitation to fall?

SQuAD (Question Answering)

In meteorology, precipitation is any product of the condensation of atmospheric water vapor that falls under gravity. The main forms of precipitation include drizzle, rain, sleet, snow, graupel and hail... Precipitation forms as smaller droplets coalesce via collision with other rain drops or ice crystals within a cloud. Short, intense periods of rain in scattered locations are called “showers”.

Q: What causes precipitation to fall?

A: gravity

SQuAD (Question Answering)

In meteorology, precipitation is any product of the condensation of atmospheric water vapor that falls under gravity. The main forms of precipitation include drizzle, rain, sleet, snow, graupel and hail... Precipitation forms as smaller droplets coalesce via collision with other rain drops or ice crystals within a cloud. Short, intense periods of rain in scattered locations are called “showers”.

Q: What is another main form of precipitation besides drizzle, rain, snow, sleet and hail?

A: graupel

SQuAD (Question Answering)

In meteorology, precipitation is any product of the condensation of atmospheric water vapor that falls under gravity. The main forms of precipitation include drizzle, rain, sleet, snow, graupel and hail... Precipitation forms as smaller droplets coalesce via collision with other rain drops or ice crystals within a cloud. Short, intense periods of rain in scattered locations are called “showers”.

Q: Where do water droplets collide with ice crystals to form precipitation?

A: within a cloud

SQuAD (Question Answering)

In meteorology, precipitation is any product of the condensation of atmospheric water vapor that falls under **gravity**. The main forms of precipitation include drizzle, rain, sleet, snow, **graupel** and hail... Precipitation forms as smaller droplets coalesce via collision with other rain drops or ice crystals **within a cloud**. Short, intense periods of rain in scattered locations are called “showers”.

Q: What causes precipitation to fall?

A: **gravity**

Q: What is another main form of precipitation besides drizzle, rain, snow, sleet and hail?

A: **graupel**

Q: Where do water droplets collide with ice crystals to form precipitation?

A: **within a cloud**

Python in Google Colab (Python101)

<https://colab.research.google.com/drive/1FEG6DnGvwfUbeo4zJ1zTunjMqf2RkCrT>

python101.ipynb ☆

File Edit View Insert Runtime Tools Help All changes saved

Comment Share Settings A

+ Code + Text

Question Answering

RAM Disk

Editing

```
[12] 1 from transformers import pipeline
2 qamodel = pipeline("question-answering", model='deepset/roberta-base-squad2')
3 question = "What causes precipitation to fall?"
4 context = """In meteorology, precipitation is any product of the condensation of atmospheric water vapor that falls under gravi·
5 output = qamodel(question = question, context = context)
6 print(output['answer'])
```

gravity

```
[13] 1 from transformers import pipeline
2 qamodel = pipeline("question-answering", model='deepset/roberta-base-squad2')
3 question = "What is another main form of precipitation besides drizzle, rain, snow, sleet and hail?"
4 context = """In meteorology, precipitation is any product of the condensation of atmospheric water vapor that falls under gravi·
5 output = qamodel(question = question, context = context)
6 print(output['answer'])
```

graupel

```
1 #from transformers import pipeline
2 #qamodel = pipeline("question-answering", model='deepset/roberta-base-squad2')
3 question = "Where do water droplets collide with ice crystals to form precipitation?"
4 context = """In meteorology, precipitation is any product of the condensation of atmospheric water vapor that falls under gravi·
5 output = qamodel(question = question, context = context)
6 print(output['answer'])
```

within a cloud

<https://tinyurl.com/aintpupython101>

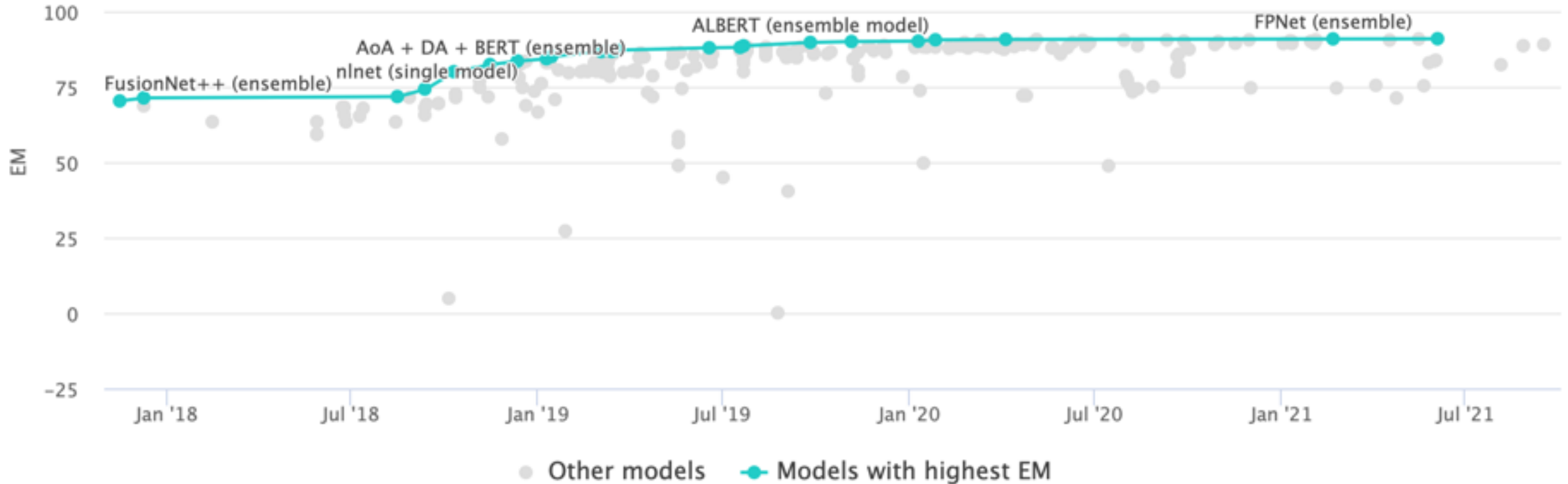
Question Answering

```
from transformers import pipeline
qamodel = pipeline("question-answering", model='deepset/roberta-base-squad2')
question = "What causes precipitation to fall?"
context = """In meteorology, precipitation is any product of
the condensation of atmospheric water vapor that falls under
gravity. The main forms of precipitation include drizzle,
rain, sleet, snow, graupel and hail... Precipitation forms as
smaller droplets coalesce via collision with other rain drops
or ice crystals within a cloud. Short, intense periods of
rain in scattered locations are called "showers"."""
output = qamodel(question = question, context = context)
print(output['answer'])
```

gravity

Question Answering on SQuAD2.0

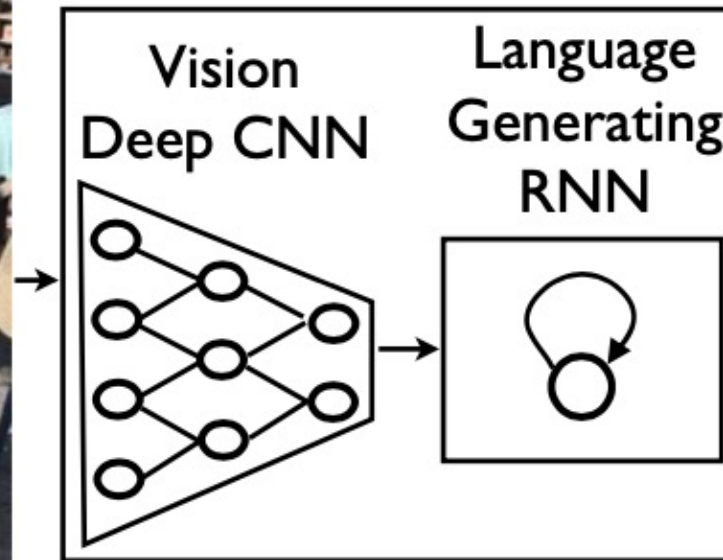
SQuAD 2.0 benchmark (Papers with Code)



<https://paperswithcode.com/sota/question-answering-on-squad20>

Neural Image Captioning (NIC)

image-to-text description generation

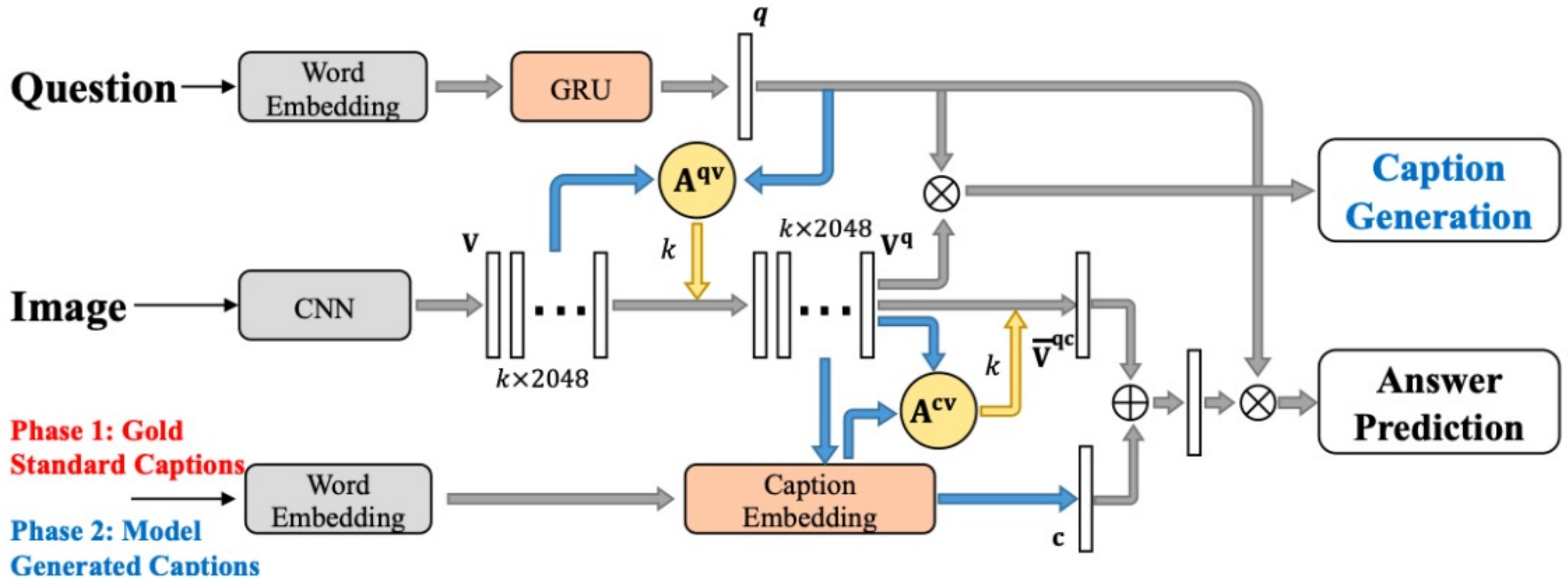


A group of people shopping at an outdoor market.

There are many vegetables at the fruit stand.

Visual Question Answering

Neural caption generation is employed to aid answer prediction



Dialogue Systems

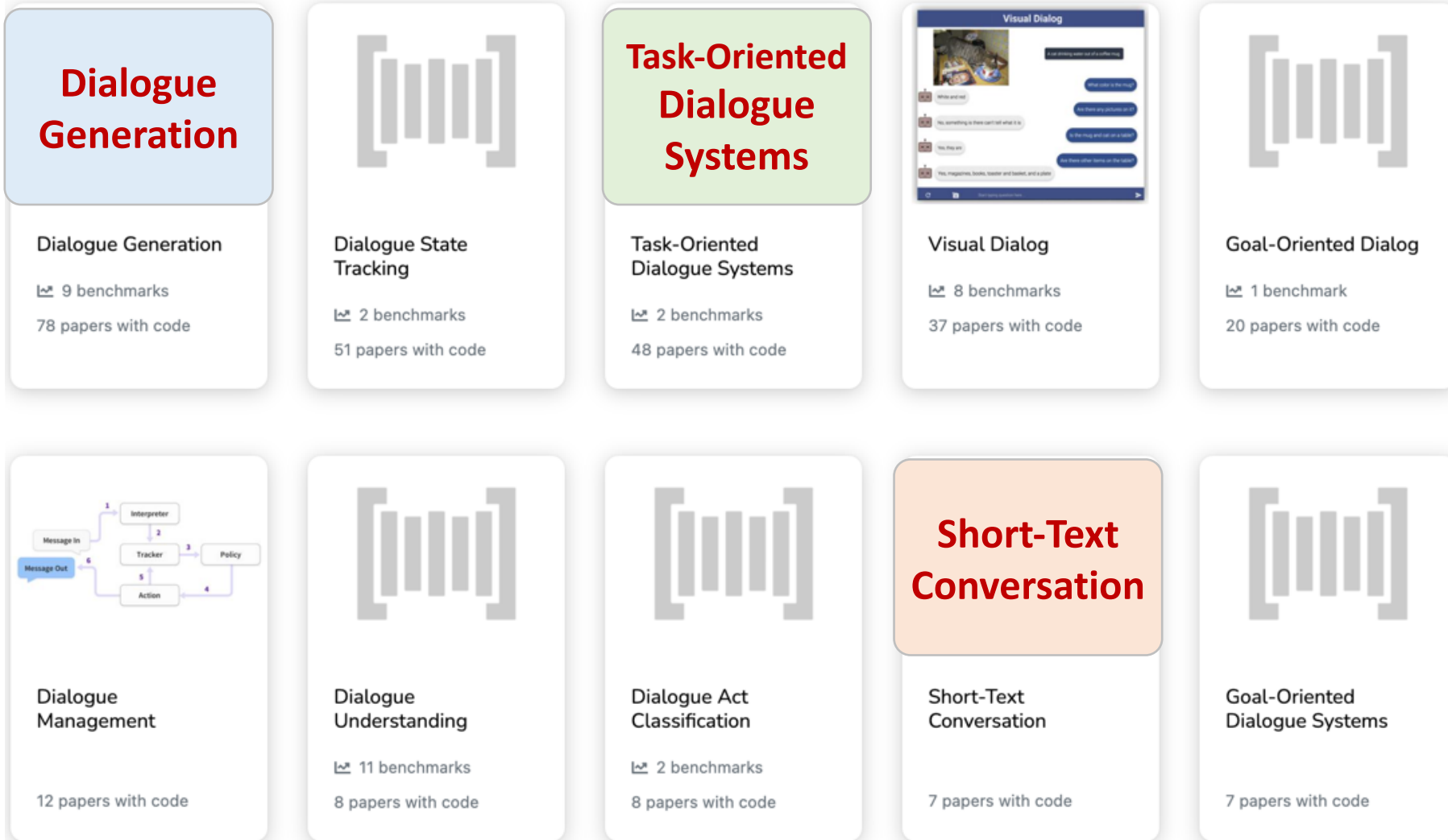
Conversational Commerce

Chatbot
Dialogue System
Intelligent Agent

Dialogue Subtasks

Browse SoTA > Natural Language Processing > Dialogue

Dialogue subtasks

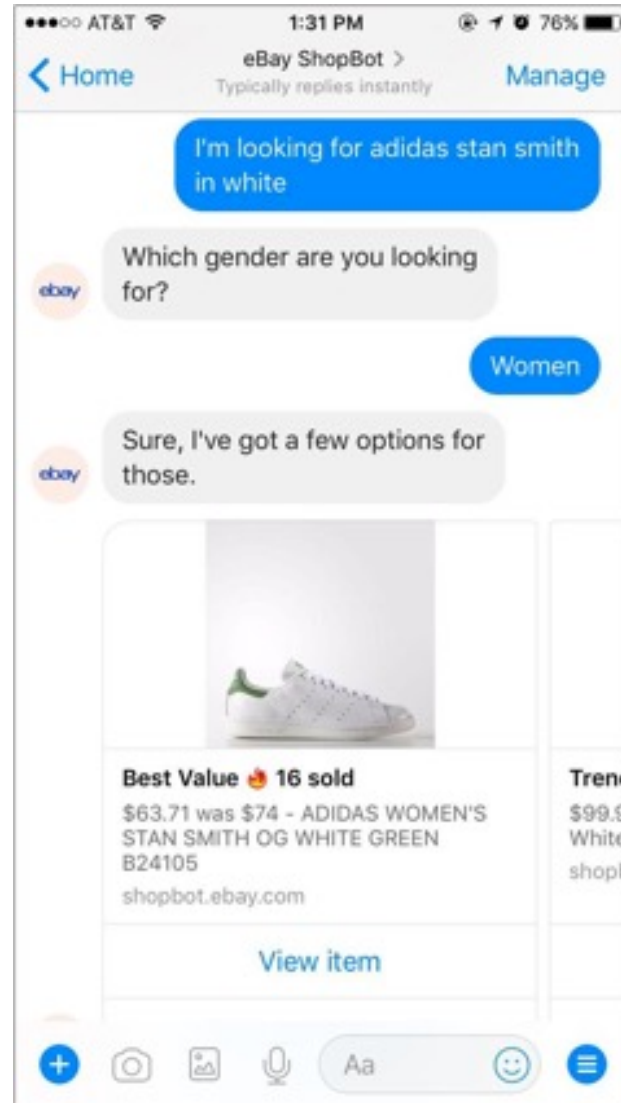


Chatbots: Evolution of UI/UX

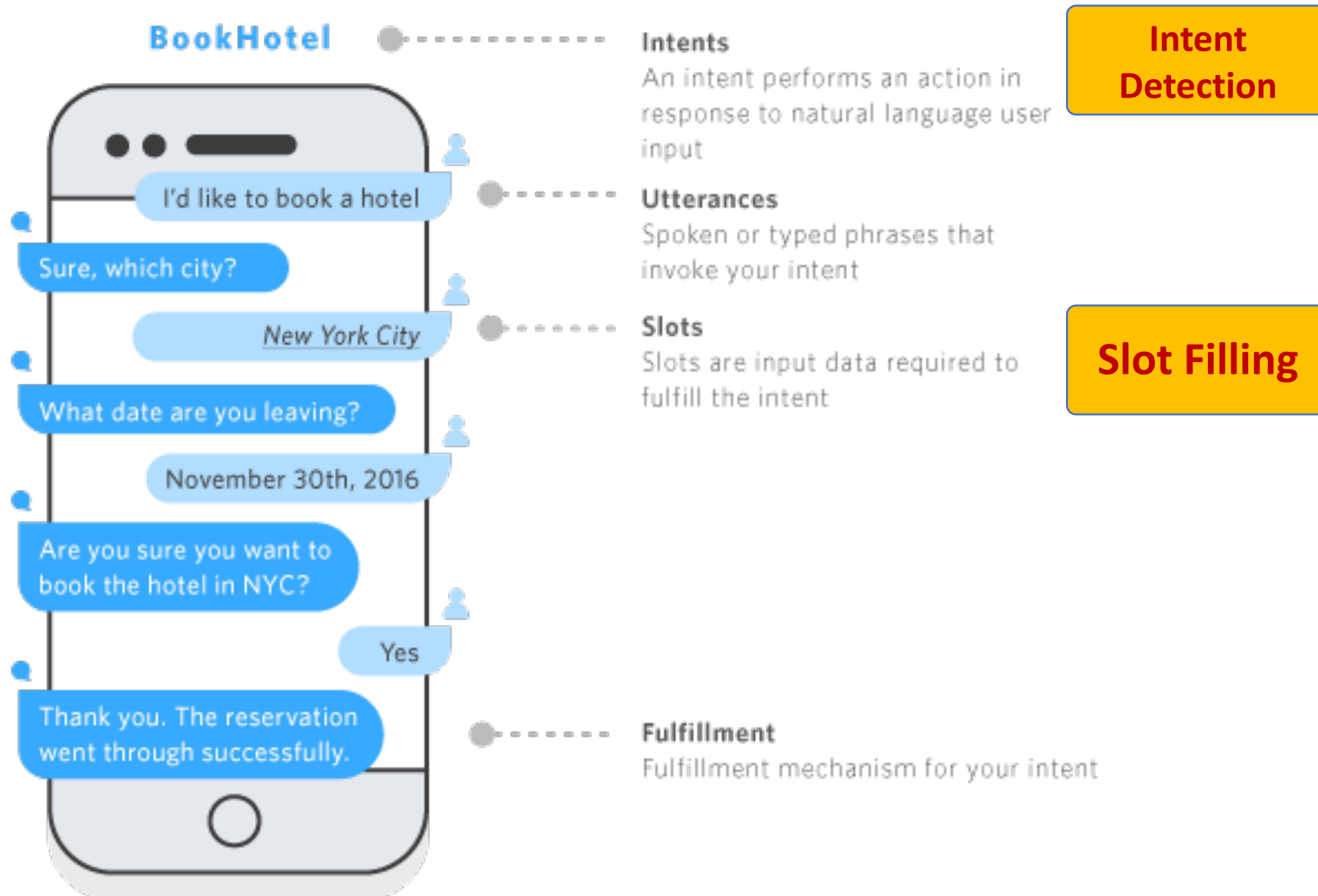
Paradigm	mid - 80s PC	mid - 90s Web	mid - 00s Smartphone	mid - 10s Messaging
Platform <i>Examples</i>	Desktop DOS, Windows, Mac OS	Browser Mosaic, Explorer, Chrome	Mobile OS iOS, Android	Messaging Apps WhatsApp, Messenger, Slack
Applications <i>Examples</i>	Clients Excel, PPT, Lotus	Website Yahoo, Amazon	Apps Angry Birds, Instagram	Bots Weather, Travel
UI/UX	Native Screens	Web Pages	Native Mobile Screens	Message
S/w Dev	Client-side	Server-side	Client-side	Server-side

**From
E-Commerce
to
Conversational Commerce:
Chatbots
and
Virtual Assistants**

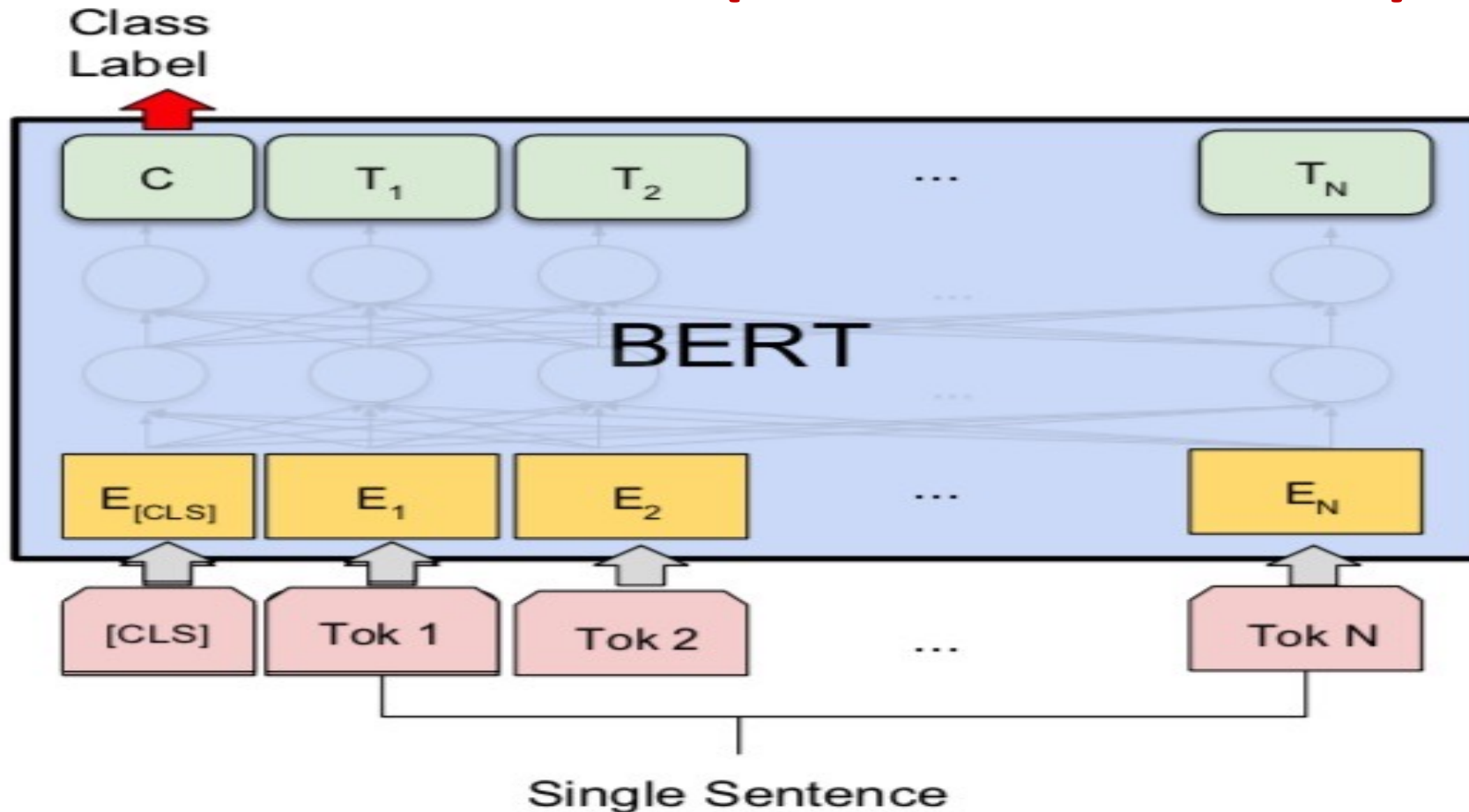
Conversational Commerce: eBay AI Chatbots



Hotel Chatbot



Fine-tuning BERT on Dialogue Intent Detection (ID; Classification)



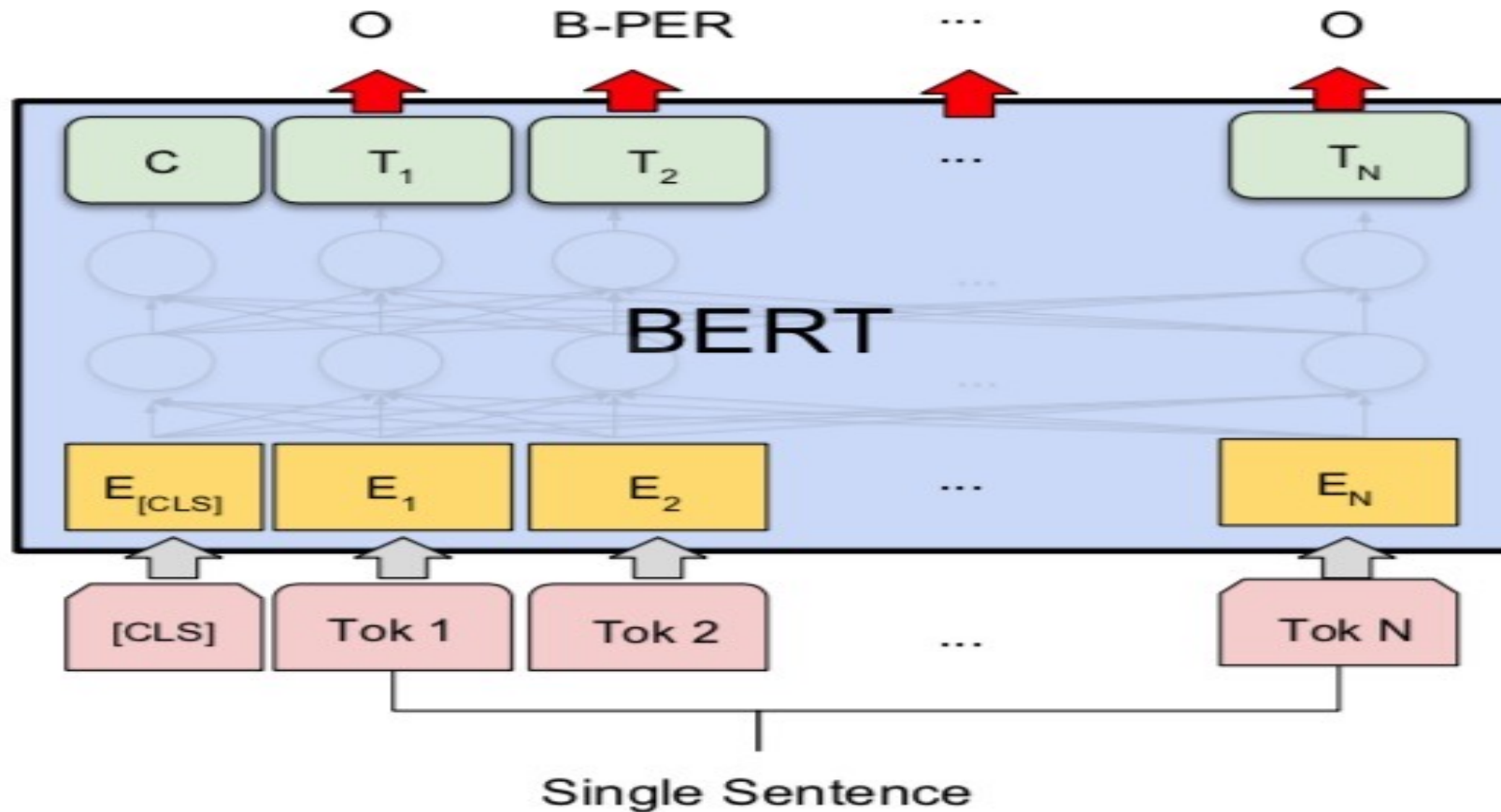
(b) Single Sentence Classification Tasks: SST-2, CoLA

Source: Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova (2018).

"Bert: Pre-training of deep bidirectional transformers for language understanding." arXiv preprint arXiv:1810.04805.

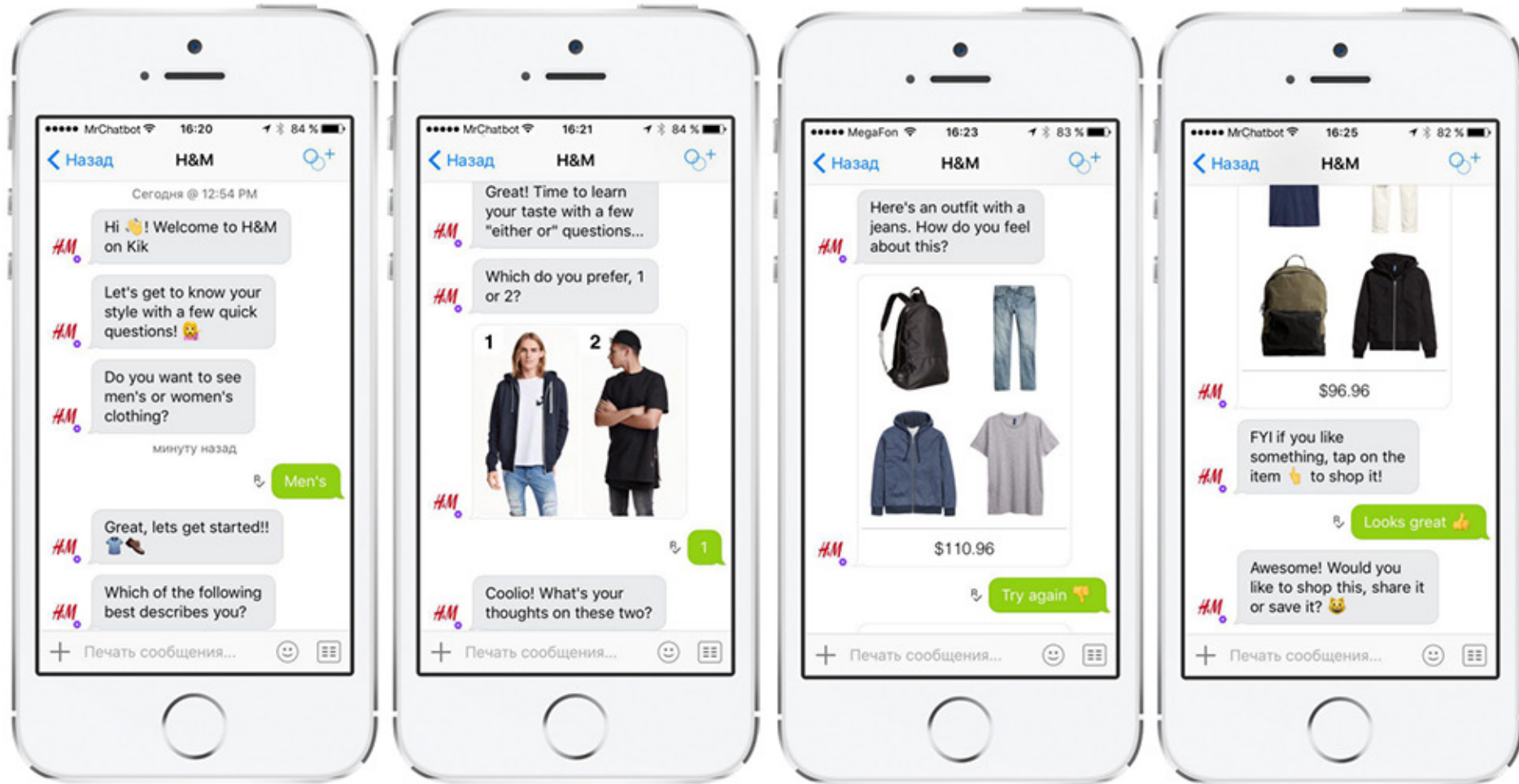
Fine-tuning BERT on Dialogue

Slot Filling (SF)

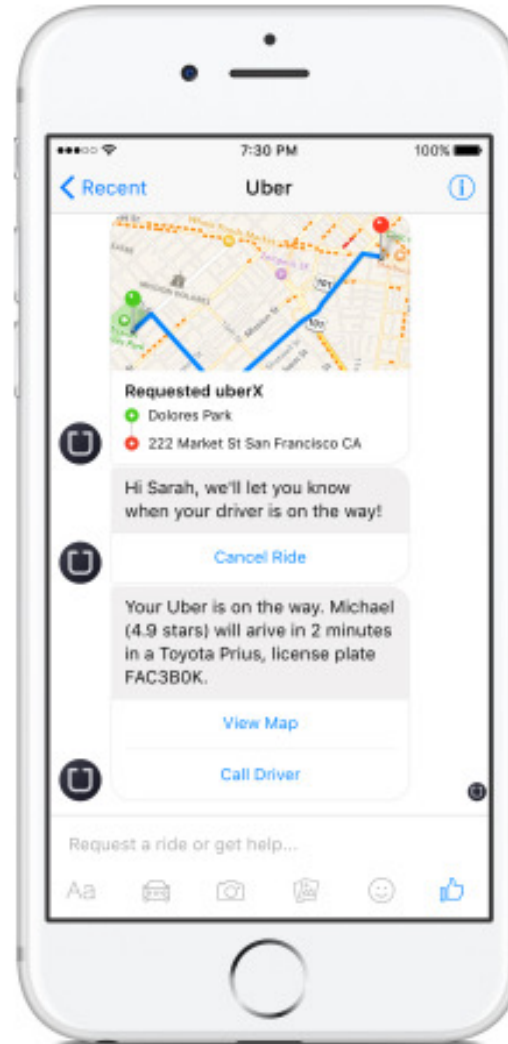


(d) Single Sentence Tagging Tasks:
CoNLL-2003 NER

H&M's Chatbot on Kik



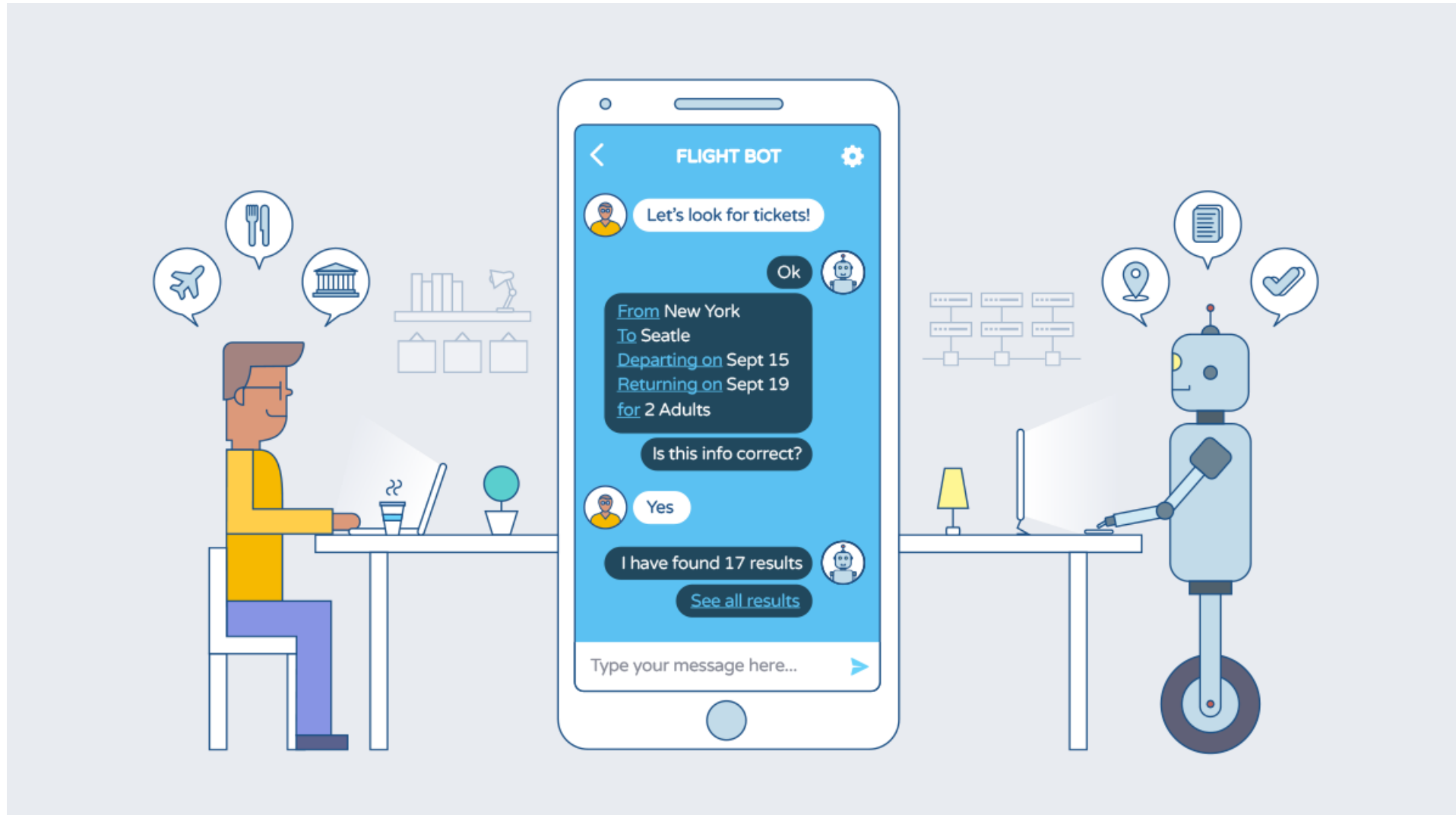
Uber's Chatbot on Facebook's Messenger



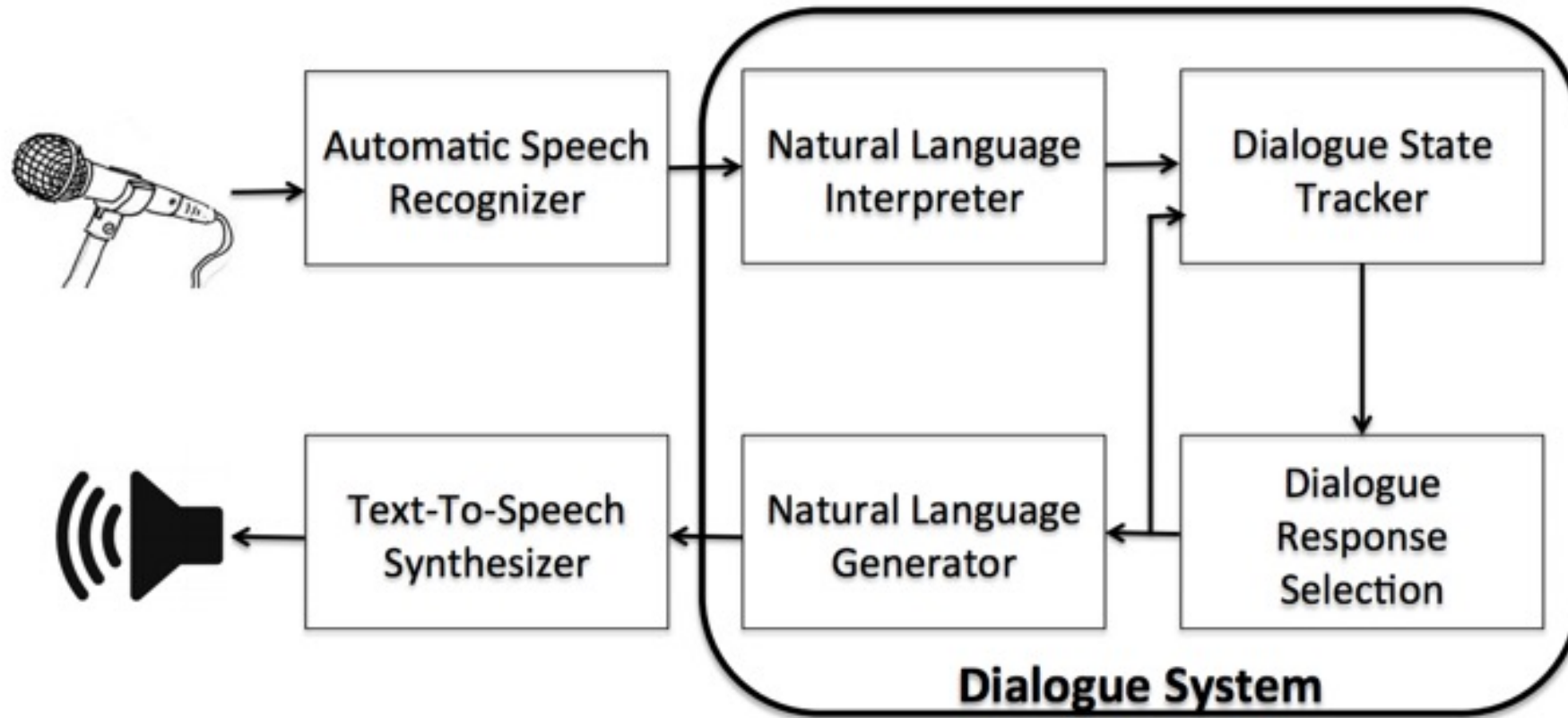
Uber's chatbot on Facebook's messenger
- one main benefit: it loads much faster than the Uber app

Source: <http://www.guided-selling.org/from-e-commerce-to-conversational-commerce/>

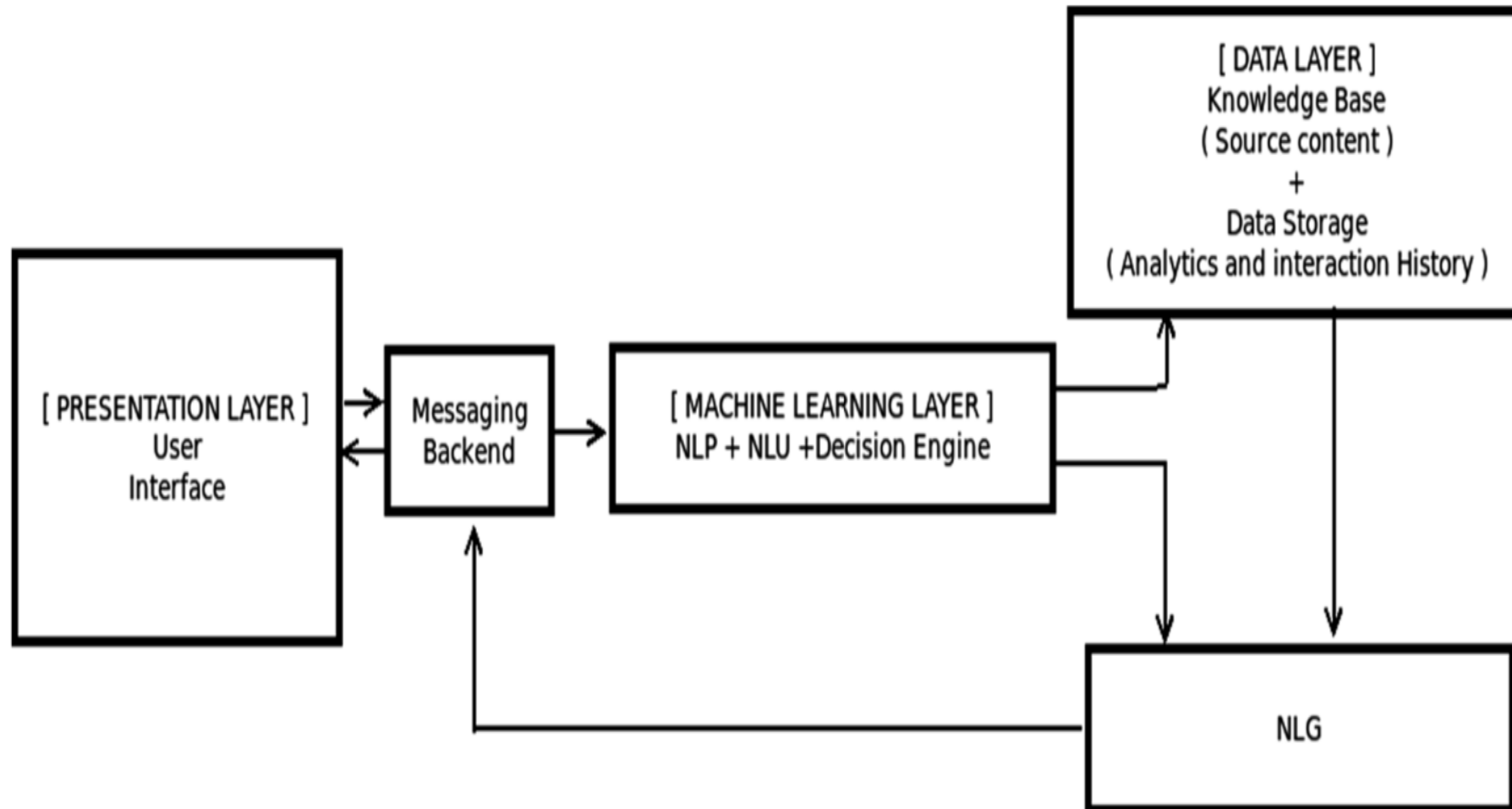
Chatbot



Dialogue System



Overall Architecture of Intelligent Chatbot



Can machines think?

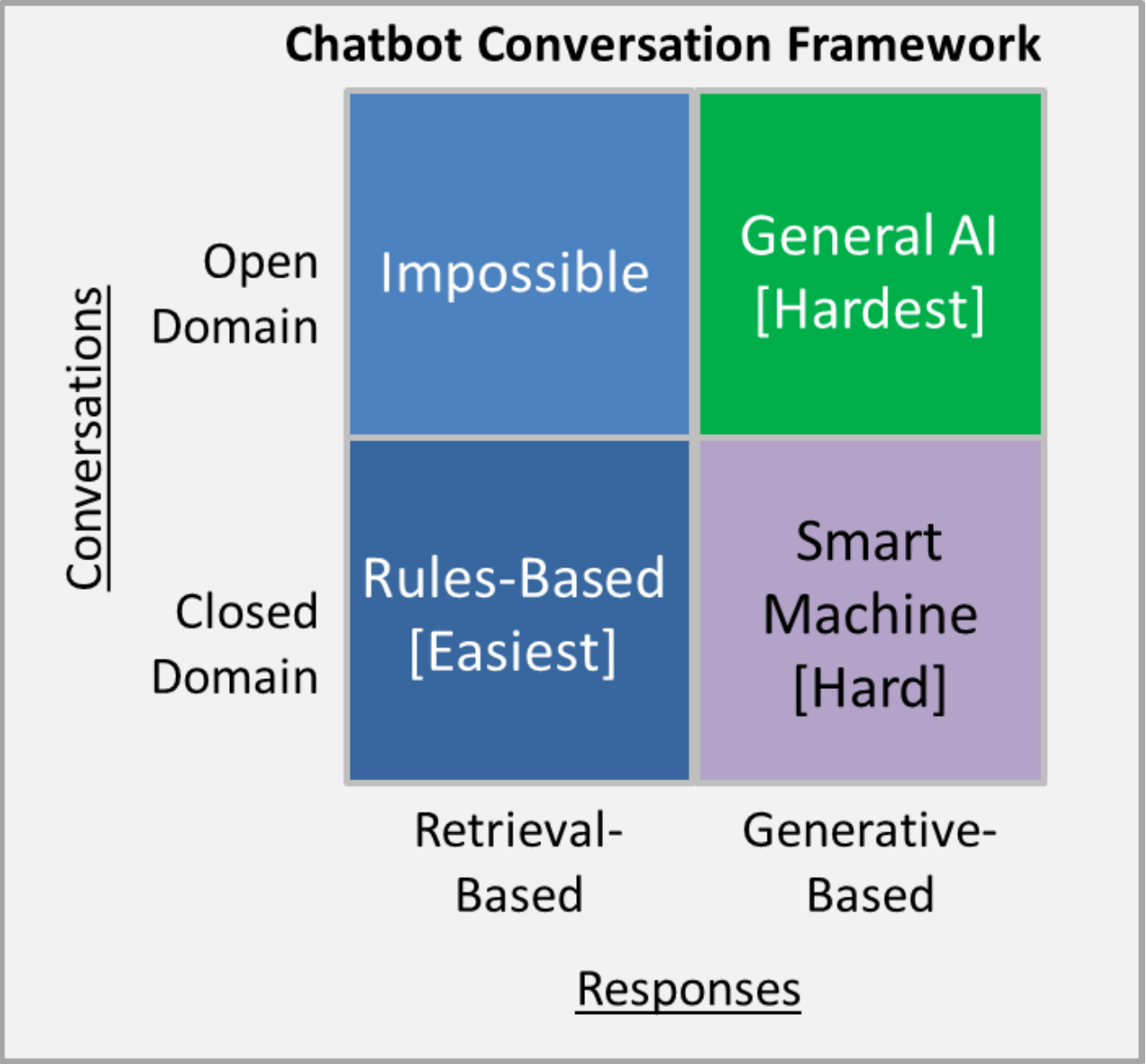
(Alan Turing ,1950)

Source: Cahn, Jack. "CHATBOT: Architecture, Design, & Development."
PhD diss., University of Pennsylvania, 2017.

Chatbot

“online human-computer
dialog system
with
natural language.”

Chatbot Conversation Framework

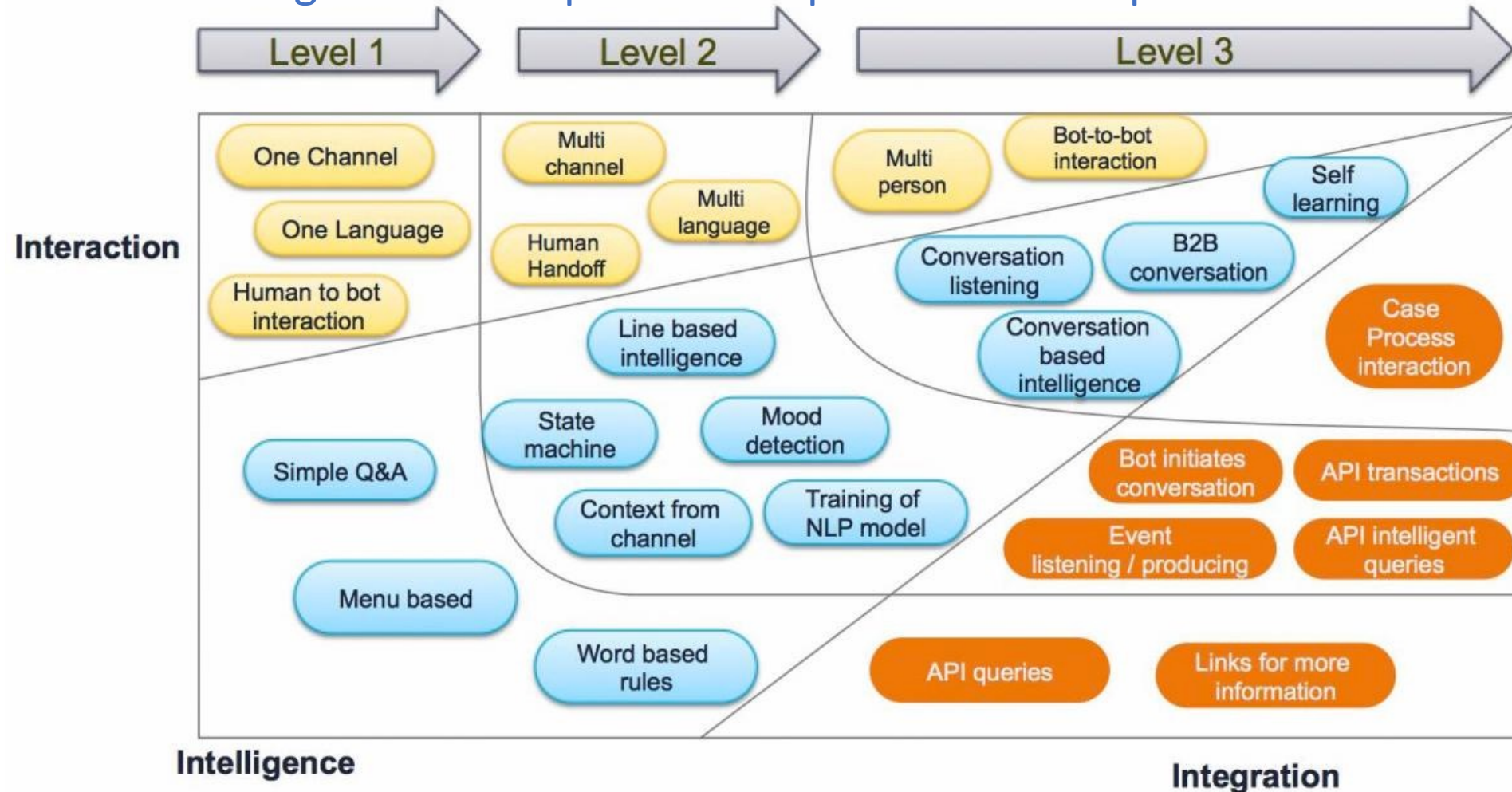


Source: <https://chatbotslife.com/ultimate-guide-to-leveraging-nlp-machine-learning-for-your-chatbot-531ff2dd870c>

Chatbots

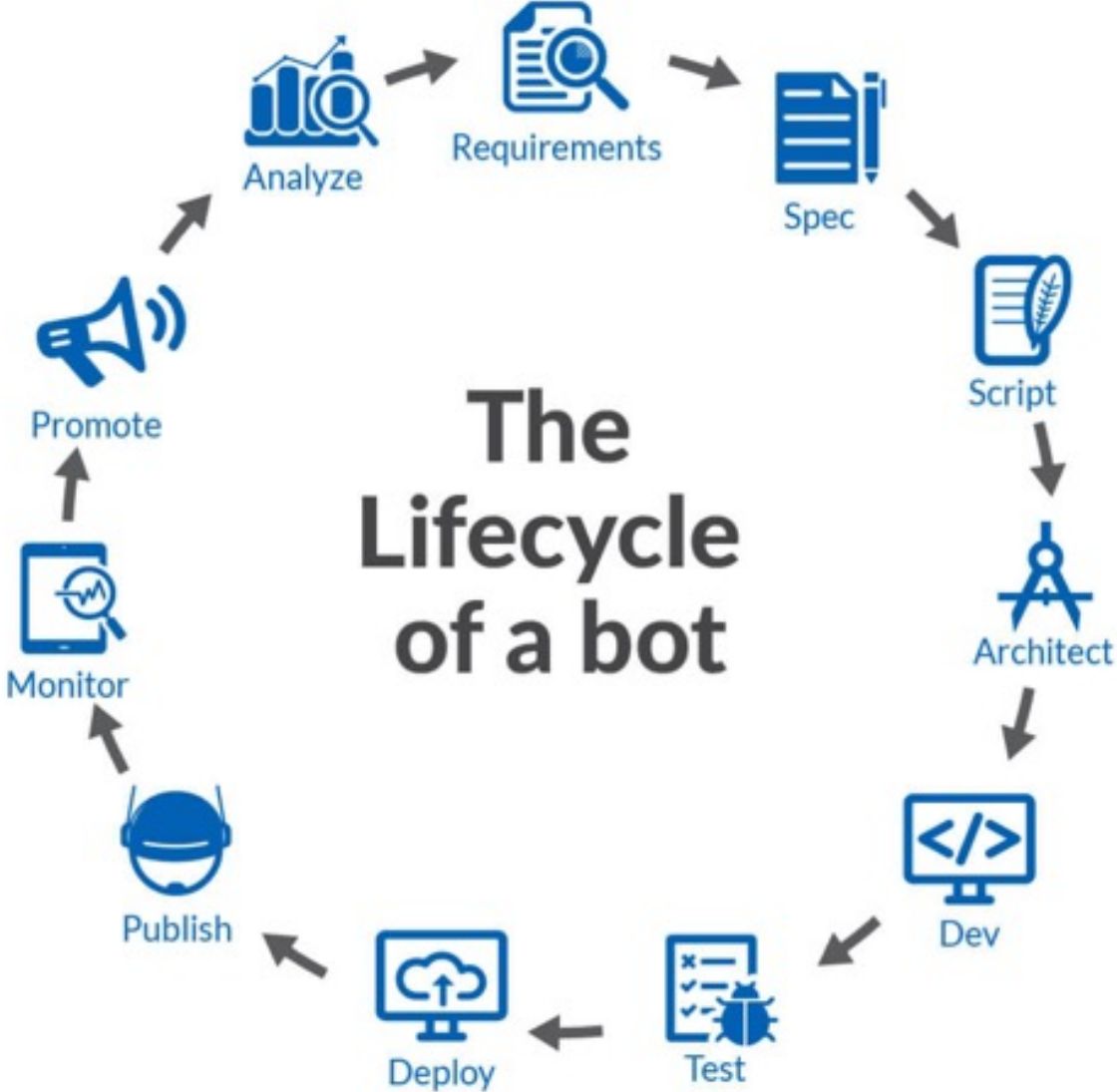
Bot Maturity Model

Customers want to have simpler means to interact with businesses and get faster response to a question or complaint.



Bot Life Cycle and Platform Ecosystem

The Bot Lifecycle



Source: <https://chatbotsmagazine.com/the-bot-lifecycle-1ff357430db7>

The bot platform ecosystem and the emerging giants

Nearly every large software company has announced some sort of bot strategy in the last year. Here's a look at a handful of leading platforms that developers might use to send messages, interpret natural language, and deploy bots, with the emerging bot-ecosystem giants highlighted.



Bot frameworks and deployment platforms



Wit.ai
Facebook



BotKit
Howdy



Chatfuel

AUTOMAT

Automat



Bot Framework
Microsoft



Api.ai
Google



Pandorabots



MindMeld

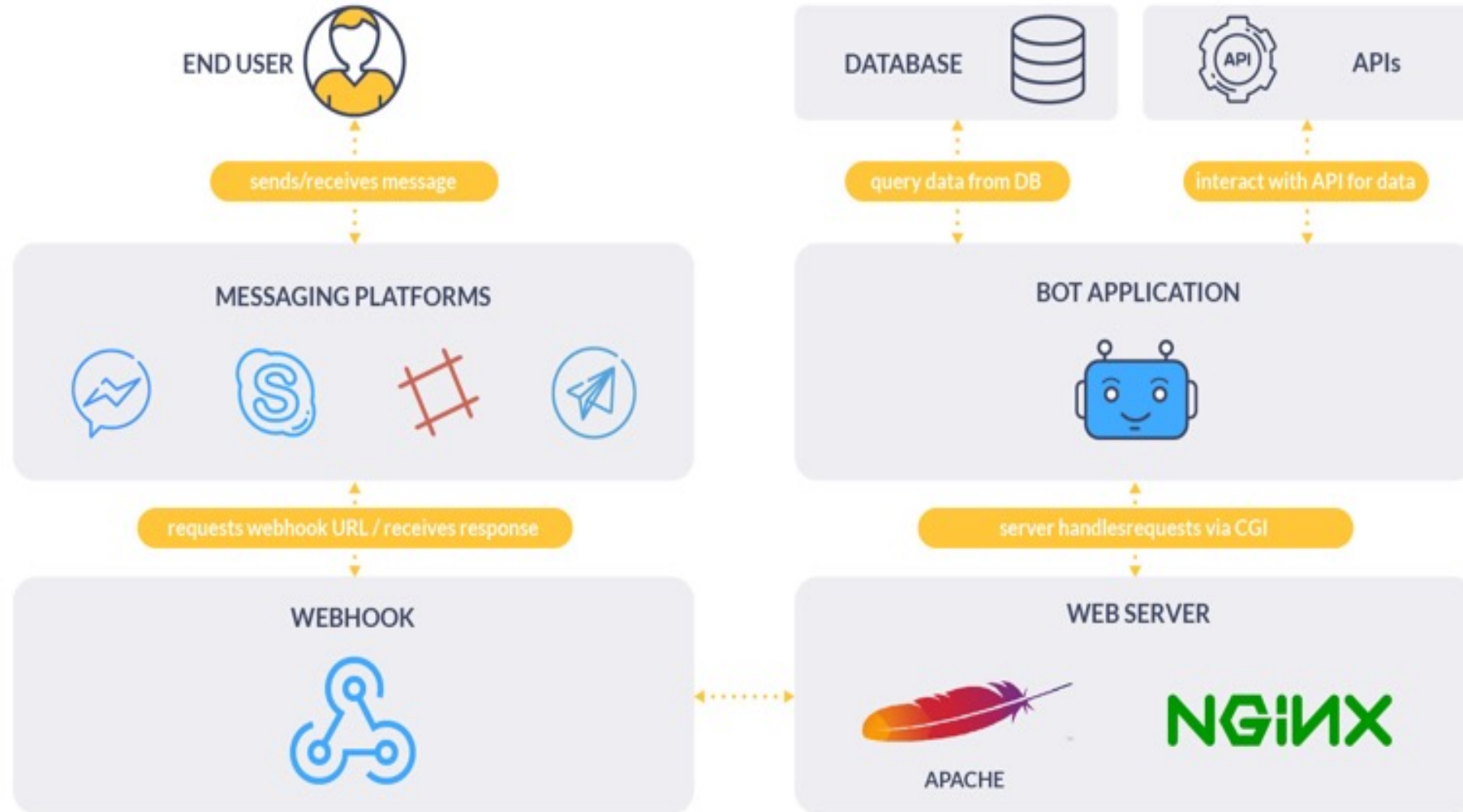


Gupshup



Sequel

How to Build Chatbots



Chatbot Frameworks and AI Services

- **Bot Frameworks**
 - **Botkit**
 - **Microsoft Bot Framework**
 - **Rasa NLU**
- **AI Services**
 - **Wit.ai**
 - **api.ai**
 - **LUIS.ai**
 - **IBM Watson**

Chatbot Frameworks

Comparison Table of Most Prominent Bot Frameworks



	Botkit	Microsoft Bot Framework	RASA NLU
Built-in Integration with messaging platforms	✓	✓	✗
NLP support	✗ but possible to integrate with middlewares	✗ but have close bonds with LUIS.ai	✓
Out-of-box bots ready to be deployed	✓	✗	✗
Programming Language	JavaScript (Node)	JavaScript (Node), C#	Python

Created by ActiveWizards

Comparison of Most Prominent AI Services

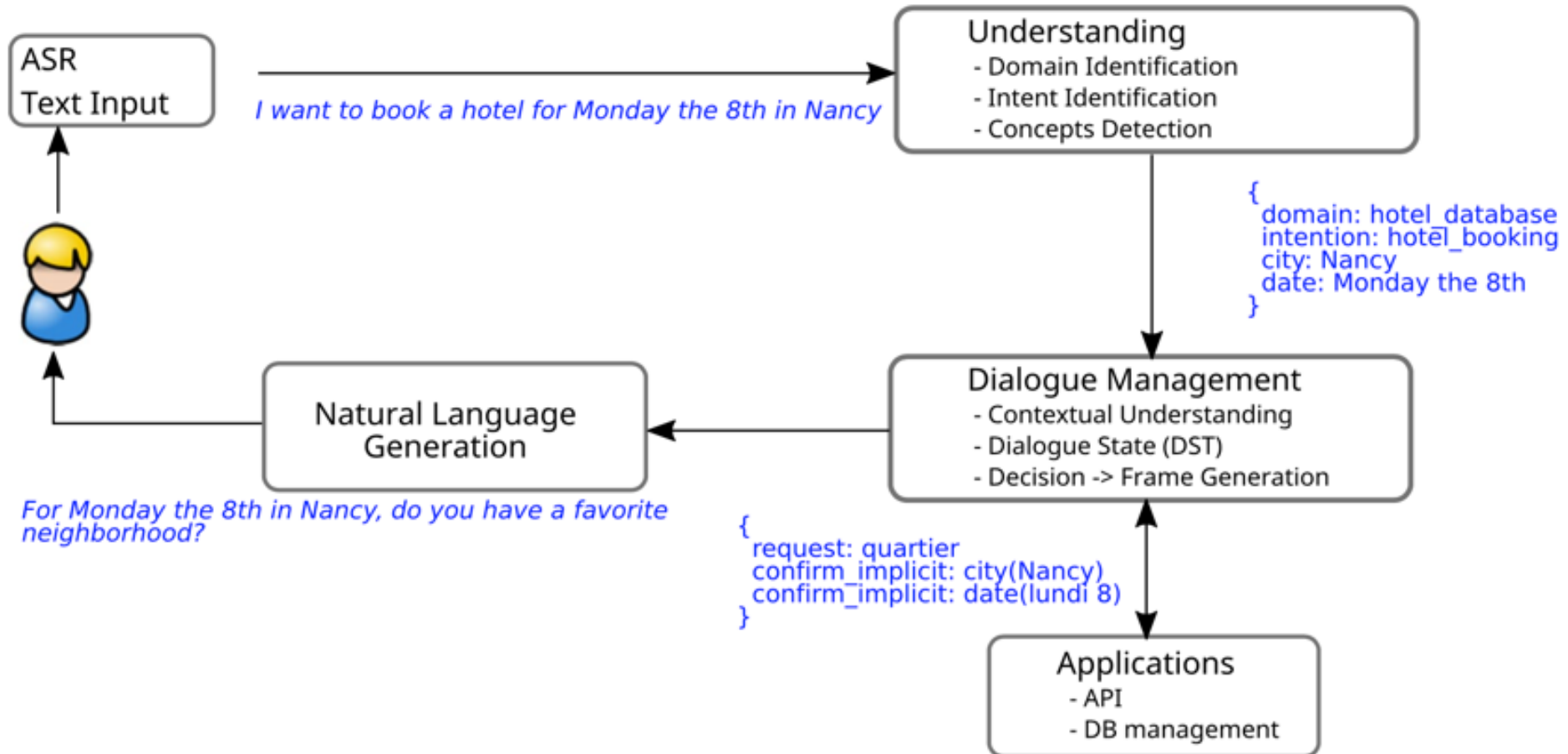
	wit.ai	api.ai	LUIS.ai	IBM Watson
Free of charge	✓	✓ but has paid enterprise version	✓ it is in beta and has transaction limits	30 days trial then priced for enterprise use
Text and Speech processing	✓	✓	✓ with use of Cortana	✓
Machine Learning Modeling	✓	✓	✓	✓
Support for Intents, Entities, Actions	✓ Intents used as trait entities, actions are combined operations	✓ Intents is the main prediction mechanism. Domains of entities, intents and actions	✓	✓
Pre-build entities for easy parsing of numbers, temperature, date, etc.	✓	✓	✓	✓
Integration to messaging platforms	✗ web service API	✓ also has facility for deploying to heroku. Paid environment	✓ integrated to Azure	✓ possible via API
Support of SDKs	✓ includes SDKs for Python, Node.js, Rust, C, Ruby, iOS, Android, Windows Phone	✓ C#, Xamarin, Python, Node.js, iOS, Android, Windows Phone	✓ enables building with Web Service API, Microsoft Bot Framework integration	Proprietary language "AlchemyLanguage"

Created by ActiveWizards

Task-Oriented Dialogue System

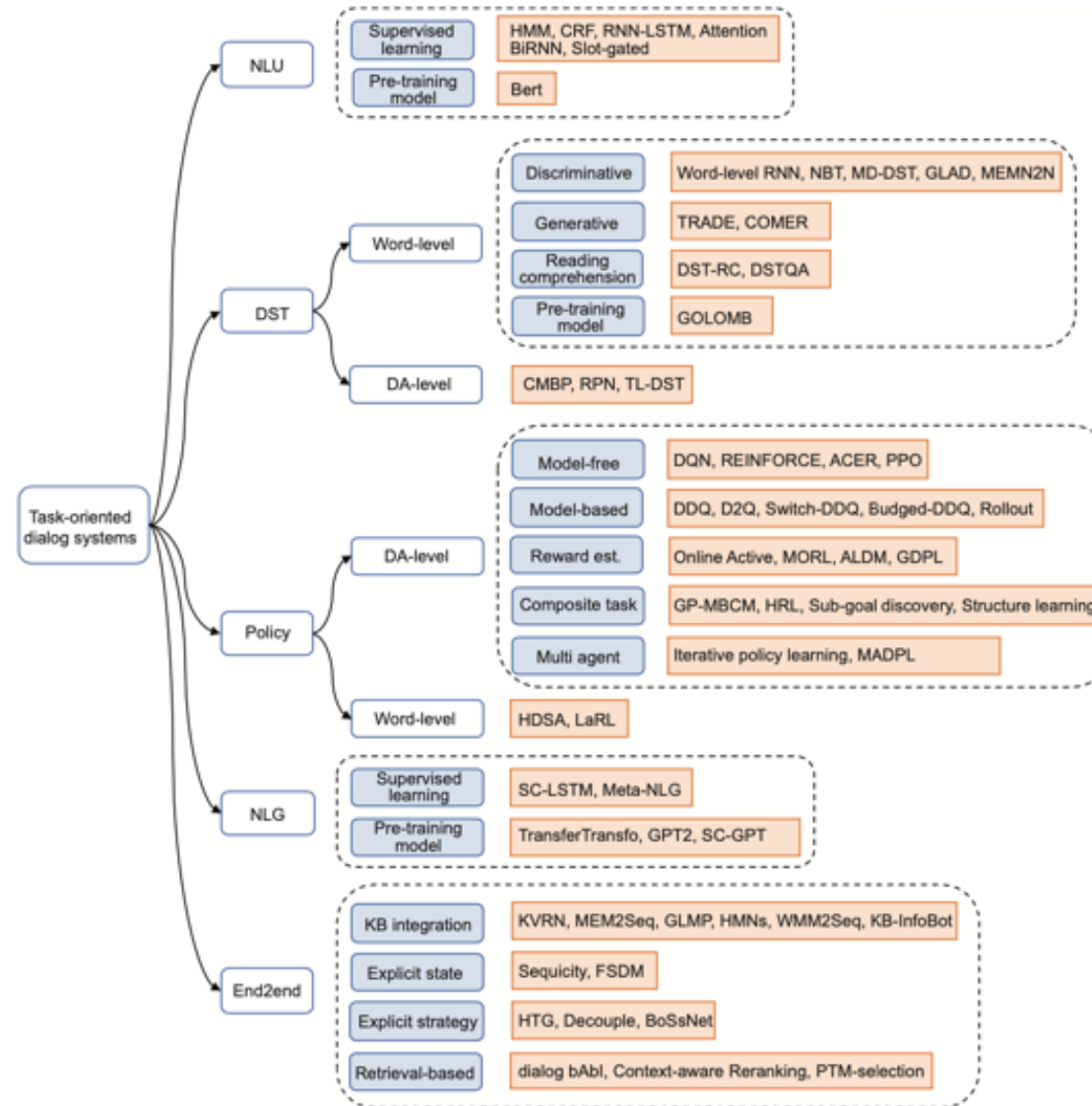
Task-Oriented Dialogue System

(Deriu et al., 2021)



Task-Oriented Dialogue Systems

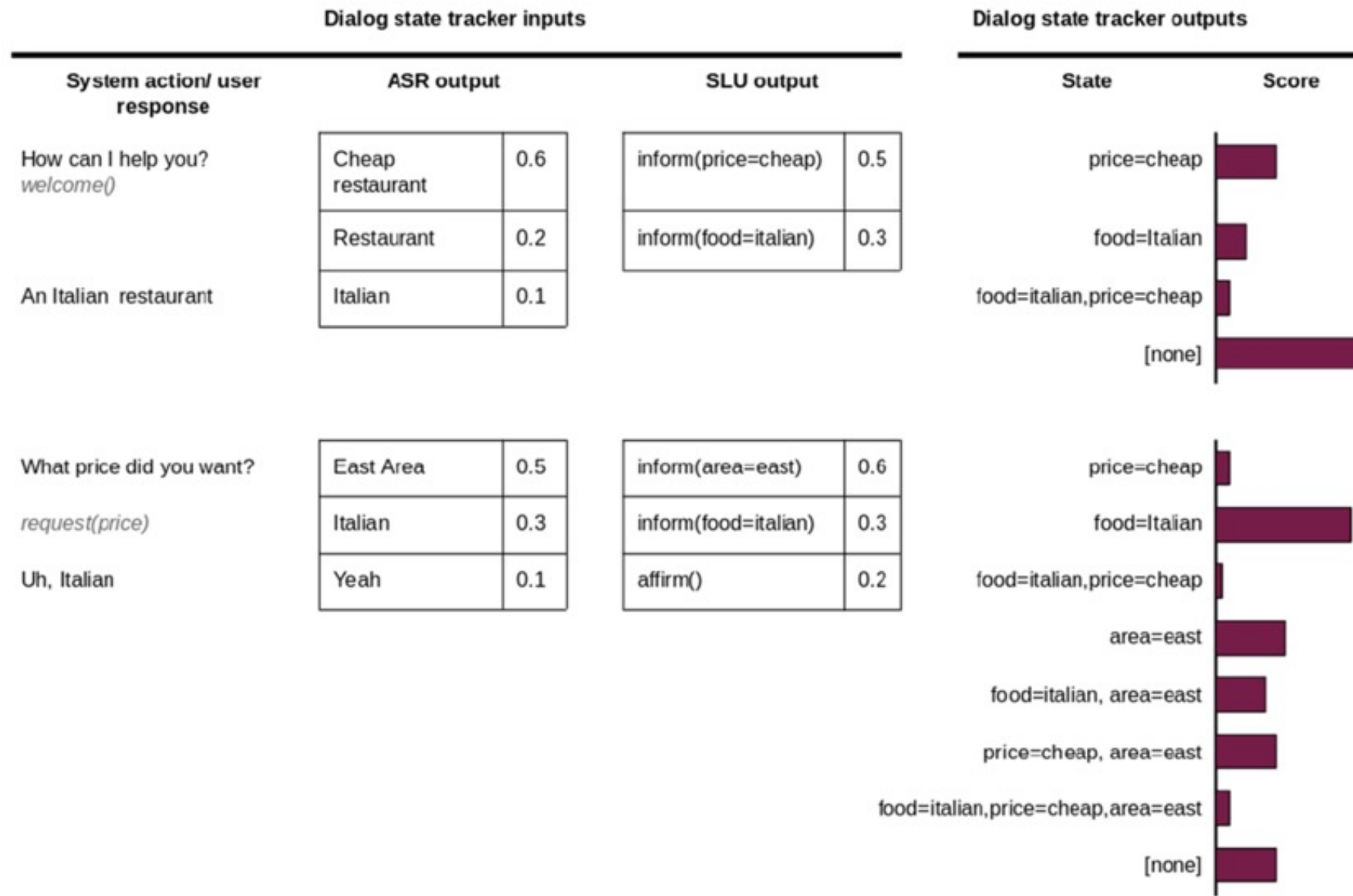
(Zhang et al., 2020)



Source: Zhang, Zheng, Ryuichi Takanobu, Qi Zhu, Minlie Huang, and Xiaoyan Zhu (2020).

"Recent advances and challenges in task-oriented dialog systems." Science China Technological Sciences (2020): 1-17.

Dialog State Tracker (DST)



Dialogue Acts

(Young et al., 2010)

Dialogue act	Description
hello($a = x, b = y, \dots$)	Open a dialogue and give info $a = x, b = y, \dots$
inform($a = x, b = y, \dots$)	Give information $a = x, b = y, \dots$
request($a, b = x, \dots$)	Request value for a given $b = x, \dots$
reqalts($a = x, \dots$)	Request alternative with $a = x, \dots$
confirm($a = x, b = y, \dots$)	Explicitly confirm $a = x, b = y, \dots$
confreq($a = x, \dots, d$)	Implicitly confirm $a = x, \dots$ and request value of d
select($a = x, a = y$)	Select either $a = x$ or $a = y$
affirm($a = x, b = y$)	Affirm and give further info $a = x, b = y, \dots$
negate($a = x$)	Negate and give corrected value $a = x$
deny($a = x$)	Deny that $a = x$
bye()	Close a dialogue

Sample Dialogue Acts

Utterance	Dialogue Act
U: Hi, I am looking for somewhere to eat	hello(task = find,type=restaurant)
S: You are looking for a restaurant. What type of food?	confreq(type = restaurant,food)
U: I'd like an Italian somewhere near the museum.	inform(food = Italian,near=museum)
S: Roma is a nice Italian restaurant near the museum.	inform(name = "Roma", type = restaurant, food = Italian, near = museum)
U: Is it reasonably priced?	confirm(pricerange = moderate)
S: Yes, Roma is in the moderate price range.	affirm(name = "Roma", pricerange = moderate)
U: What is the phone number?	request(phone)
S: The number of Roma is 385456.	inform(name = "Roma", phone = "385456")
U: Ok, thank you goodbye.	bye()

**Dialogue
on
Airline Travel
Information System
(ATIS)**

The ATIS (Airline Travel Information System) Dataset

<https://www.kaggle.com/siddhadev/atis-dataset-from-ms-cntk>

Sentence	what	flights	leave	from	phoenix
Slots	O	O	O	O	B-fromloc
Intent	atis_flight				

Training samples: 4978

Testing samples: 893

Vocab size: 943

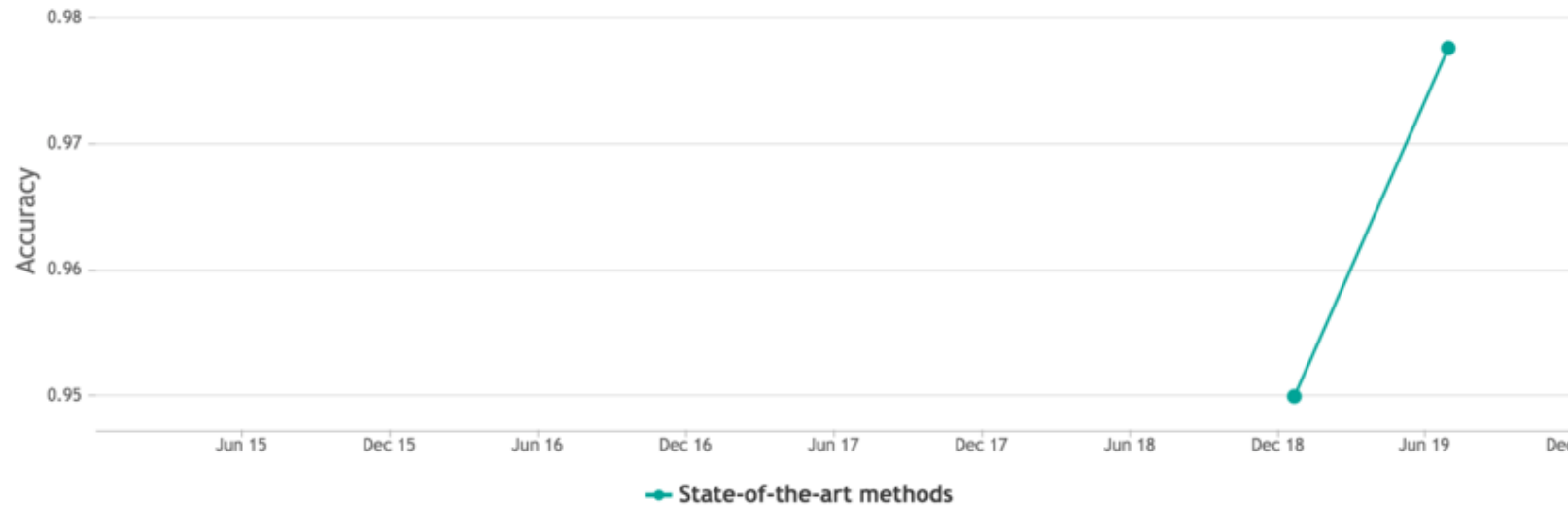
Slot count: 129

Intent count: 26

Intent Detection on ATIS

State-of-the-art

Intent Detection on ATIS



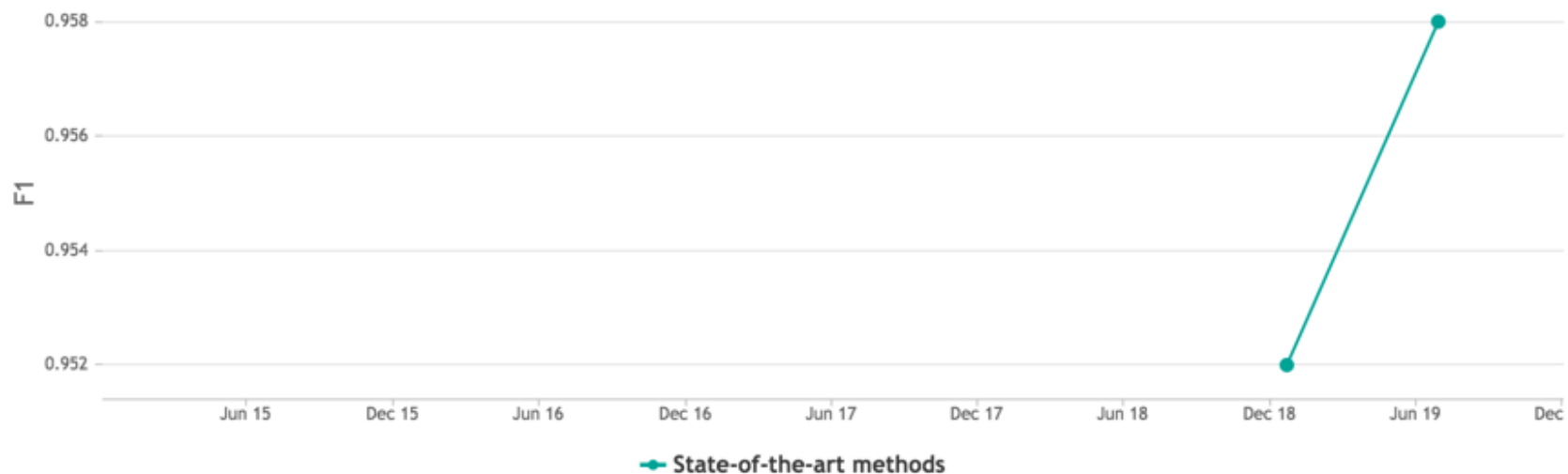
[Edit](#)

RANK	METHOD	ACCURACY	PAPER TITLE	YEAR	PAPER	CODE
1	SF-ID	0.9776	A Novel Bi-directional Interrelated Model for Joint Intent Detection and Slot Filling	2019		
2	Capsule-NLU	0.950	Joint Slot Filling and Intent Detection via Capsule Neural Networks	2018		

Slot Filling on ATIS

State-of-the-art

Slot Filling on ATIS



RANK	METHOD	F1	PAPER TITLE	YEAR	PAPER	CODE
1	SF-ID	0.958	A Novel Bi-directional Interrelated Model for Joint Intent Detection and Slot Filling	2019		
2	Capsule-NLU	0.952	Joint Slot Filling and Intent Detection via Capsule Neural Networks	2018		

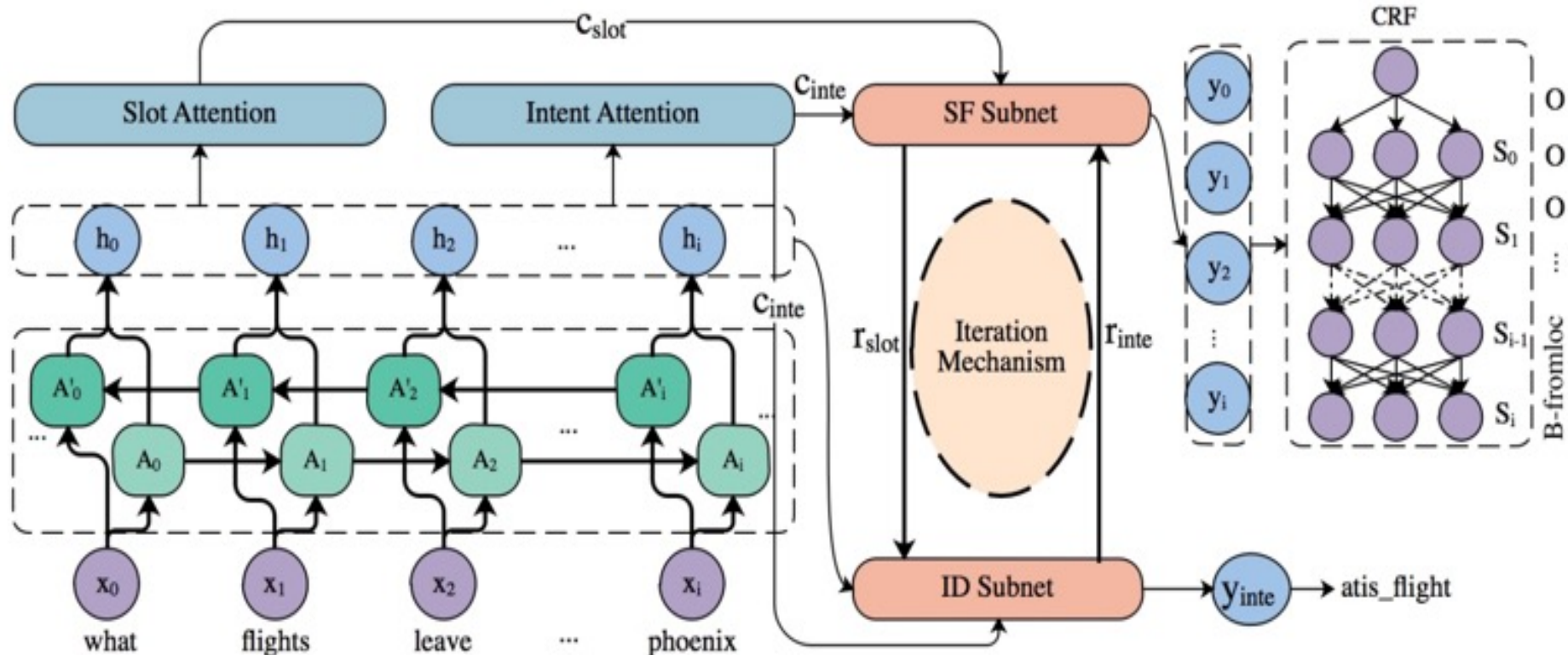
[Edit](#)

SF-ID Network (E et al., 2019)

Slot Filling (SF)

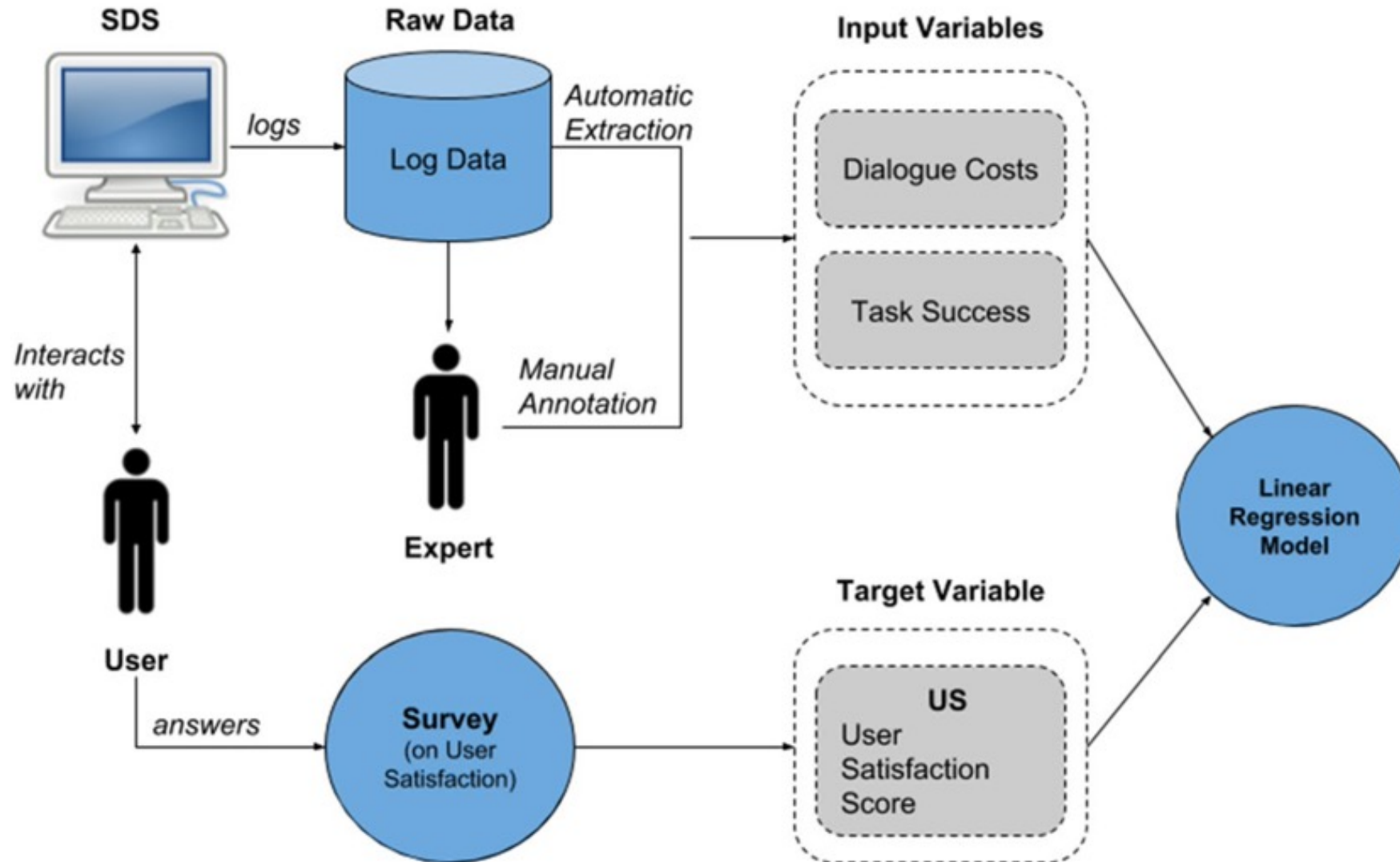
Intent Detection (ID)

A Novel Bi-directional Interrelated Model for Joint Intent Detection and Slot Filling



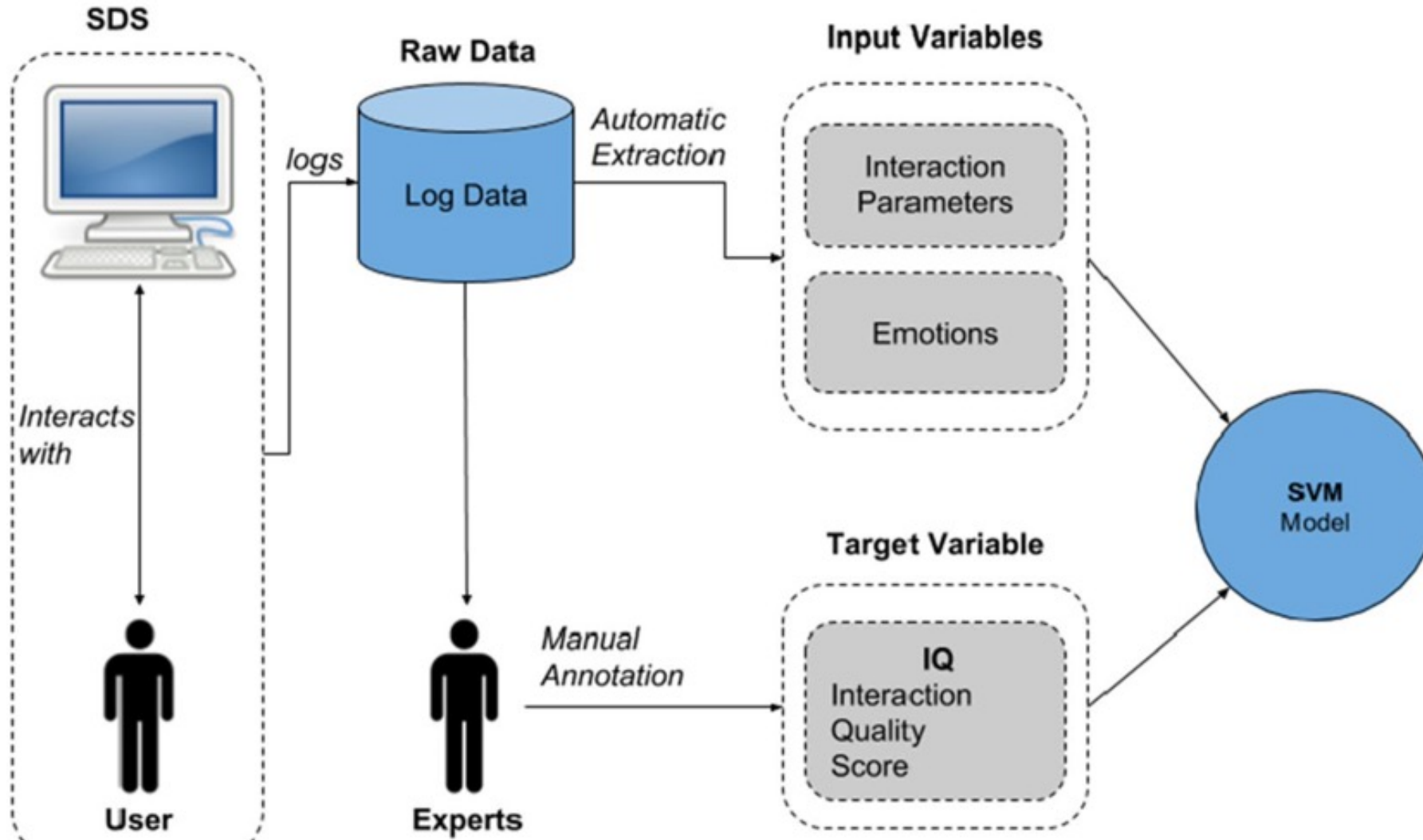
PARAdigm for Dialog System Evaluation

PARADISE Framework (Walker et al. 1997)



Interaction Quality procedure

(Schmitt and Ultes, 2015)



Datasets for task-oriented dialogue systems

Name	Topics	# dialogues	Reference
DSTC1	Bus schedules	15,000	(Williams et al. 2013)
DSTC2	Restaurants	3000	(Henderson et al. 2014)
DSTC3	Tourist information	2265	(Henderson et al. 2013a)
DSTC4 & DSTC5	Tourist information	35	(Kim et al. 2016)
DSTC6	Restaurant reservation	–	(Perez et al. 2017)
DSTC7 (Flex Data)	Student guiding	500	(Gunasekara et al. 2019)
DSTC8 (MetaLWOz)	47 domains	37,884	(Lee et al. 2019)
DSTC8 (Schema-Guided)	20 domains	22,825	(Rastogi et al. 2019)
MultiWOZ	Tourist information	10,438	(Budzianowski et al. 2018)
Taskmaster-1	6 domains	13,215	(Byrne et al. 2019)
MultiDoGo	6 domains	86,698	(Peskov et al. 2019)

Restaurants Dialogue Datasets

- **MIT Restaurant Corpus**
 - <https://groups.csail.mit.edu/sls/downloads/restaurant/>
- **CamRest676**
(Cambridge restaurant dialogue domain dataset)
 - <https://www.repository.cam.ac.uk/handle/1810/260970>
- **DSTC2 (Dialog State Tracking Challenge 2 & 3)**
 - <http://camdial.org/~mh521/dstc/>

CrossWOZ:

A Large-Scale Chinese Cross-Domain Task-Oriented Dialogue Dataset

MultiWOZ	CrossWOZ
usr: I'm looking for a college type attraction. ... usr: I would like to visit <u>in town centre</u> please. ... usr: Can you find an Indian restaurant for me that is also in the town centre ?	usr: 你好，可以帮我推荐一个评分是4.5分以上的景点吗？ Hello, could you recommend an attraction with a rating of 4.5 or higher? sys: 天安门城楼，簋街小吃和 <u>北京欢乐谷</u> 都是很不错的地方呢。 Tiananmen, Gui Street, and <u>Beijing Happy Valley</u> are very nice places. usr: 我喜欢 <u>北京欢乐谷</u> ，你知道这个景点周边的酒店都是什么吗？ I like <u>Beijing Happy Valley</u> . What hotels are around this attraction? sys: 那可多了，有A酒店，B酒店，C酒店。 There are many, such as hotel A, hotel B, and hotel C. usr: 太好了，我正打算在 <u>景点附近</u> 找个酒店住宿呢，知道哪家评分是4分以上，提供叫醒服务的不？ Great! I am planning to find a hotel to stay near the attraction . Which one has a rating of 4 or higher and offers wake-up call service?
Schema	
usr: I want a hotel in <u>San Diego</u> and I want to check out on Thursday next week. ... usr: I need a one way flight to go there .	

CrossWOZ:

A Large-Scale Chinese Cross-Domain Task-Oriented Dialogue Dataset

Type	Single-domain goal					Multi-domain goal		
Dataset	DSTC2	WOZ 2.0	Frames	KVRET	M2M	MultiWOZ	Schema	CrossWOZ
Language	EN	EN	EN	EN	EN	EN	EN	CN
Speakers	H2M	H2H	H2H	H2H	M2M	H2H	M2M	H2H
# Domains	1	1	1	3	2	7	16	5
# Dialogues	1,612	600	1,369	2,425	1,500	8,438	16,142	5,012
# Turns	23,354	4,472	19,986	12,732	14,796	115,424	329,964	84,692
Avg. domains	1	1	1	1	1	1.80	1.84	3.24
Avg. turns	14.5	7.5	14.6	5.3	9.9	13.7	20.4	16.9
# Slots	8	4	61	13	14	25	214	72
# Values	212	99	3,871	1363	138	4,510	14,139	7,871

Task-Oriented Dialogue

Initial user state (=user goal)

id=1(Attraction): fee=free,
name=?, nearby hotels=?

id=2(Hotel): **name=near (id=1)**,
wake-up call=yes, rating=?

id=3(Taxi): **from=(id=1), to=(id=2)**,
car type=? plate number=?

...

Final user state

id=1 (Attraction): name=Tiananmen Square,
fee=free, nearby hotels=[Beijing Capital
Hotel, Guidu Hotel Beijing]

id=2 (Hotel): **name=Beijing Capital Hotel**,
wake-up call=yes, rating=4.6

id=3 (Taxi): **from=Tiananmen Square**,
to=Beijing Capital Hotel,
car type=#CX, plate number=#CP



Source: Zhu, Qi, Kaili Huang, Zheng Zhang, Xiaoyan Zhu, and Minlie Huang. "Crosswoz: A large-scale chinese cross-domain task-oriented dialogue dataset." arXiv preprint arXiv:2002.11893 (2020).

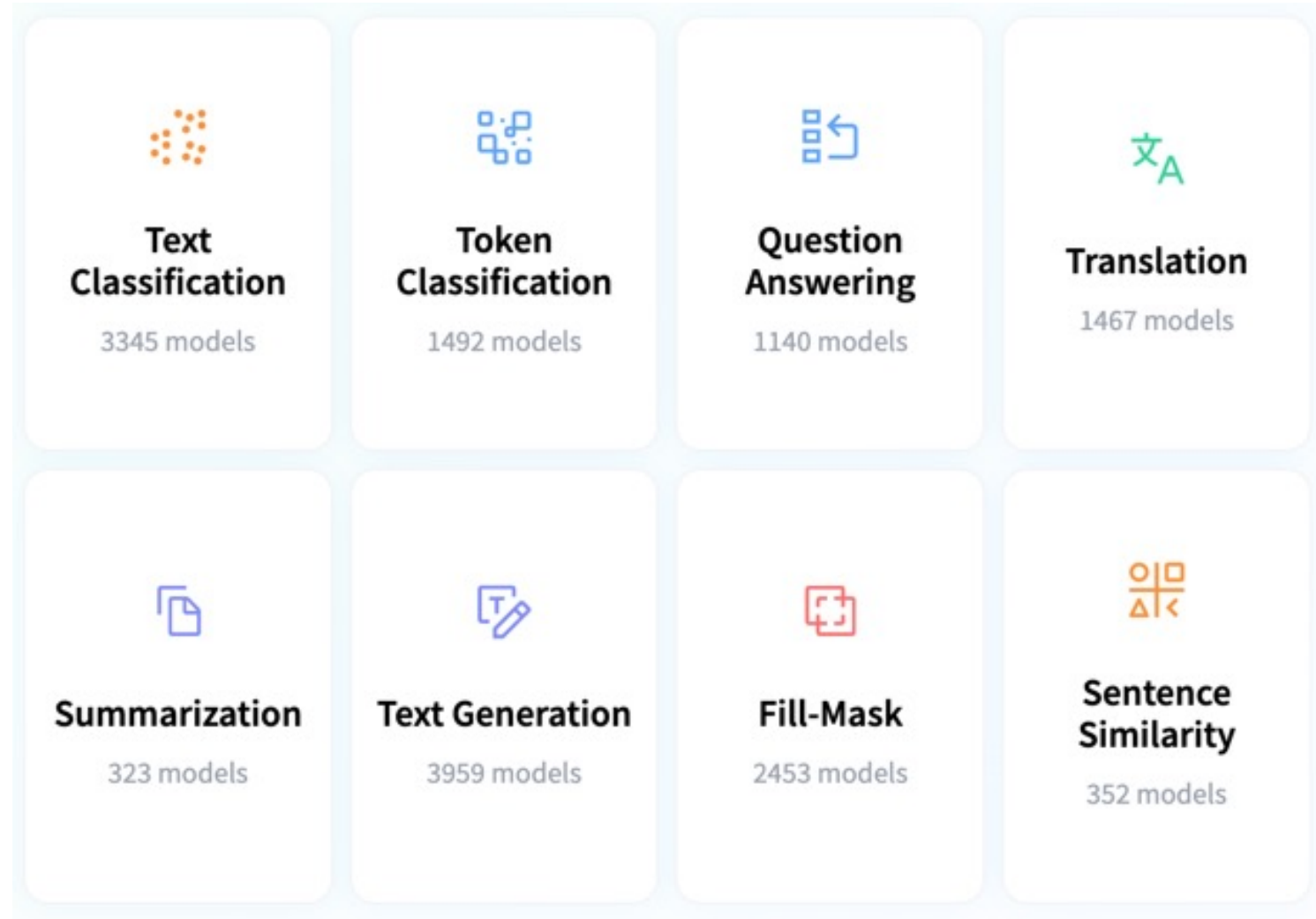
An example dialog from the test set for MultiWOZ

(en→zh) sub-task

Speaker	Utterance	Dialog State Update
User	Hello! I am looking for a local guesthouse in the centre. 你好！我在市中心找一家本地宾馆。	hotel: {area: centre, type: guesthouse}
System	OK. I am glad to recommend Alexander Bed and Breakfast to you. 好的。这边很高兴向您推荐亚历山大住宿加早餐旅馆。	旅馆: {区域: 中心, 类型: 宾馆}
User	Where is it? 它在哪里?	hotel: {name: alexander bed and breakfast}
System	It is at 56 Saint Barnabas Road. 圣巴纳巴斯路56号。	旅馆: {名称: 亚历山大住宿加早餐旅馆}
User	I also wish to have a meal in a local European restaurant in the centre. 我还想在市中心的一家本地欧洲餐厅吃饭。	restaurant: {food: european, area: centre}
System	You can choose Eraina. 您可以选择伊莱娜。	餐厅: {食物: 欧洲的, 区域: 中心}
User	Please give me its address. 那请给我它的地址。	restaurant: {name: eraina}
System	It is in Free School Lane City Centre. 市中心自由校园巷。	餐厅: {名称: 伊莱娜}
User	Ok. I'll go there. I need to book a taxi from Alexander Bed and Breakfast to Eraina after 07:00. 好。那我去那里。我还要预订一辆7:00时从亚历山大住宿加早餐旅馆到伊莱娜的出租车。	taxi: {leaveAt: 07:00, destination: eraina, departure: alexander bed and breakfast}
System	Well. I find a yellow Skoda. 好的。是一辆黄色的斯柯达。	出租车: {出发时间: 07:00, 目的地: 伊莱娜, 出发地: 亚历山大住宿加早餐旅馆}
User	How about its phone number? 它的电话号码是多少?	No update
System	It is 78519675253. 78519675253。	
User	Thank you for your help. Bye! 谢谢你帮忙。再见!	No update
System	A pleasure. Bye bye! 我很乐意。再见!	

Hugging Face Tasks

Natural Language Processing



<https://huggingface.co/tasks>

NLP with Transformers Github

The screenshot shows the GitHub repository page for 'nlp-with-transformers/notebooks'. The repository is public and has 170 forks and 1.1k stars. The main branch is 'main'. The repository contains several files and folders, including a README, a .gitignore, and several Jupyter notebooks (01_introduction.ipynb, 02_classification.ipynb, 03_transformer-anatomy.ipynb, 04_multilingual-ner.ipynb, 05_text-generation.ipynb). The repository is described as 'Jupyter notebooks for the Natural Language Processing with Transformers book' and is licensed under Apache-2.0. The repository is also linked to the book 'Natural Language Processing with Transformers' by Lewis Tunstall, Leandro van Werra, and Thomas Wolf.

Why GitHub? Team Enterprise Explore Marketplace Pricing

Search / Sign in Sign up

nlp-with-transformers / notebooks Public

Notifications Fork 170 Star 1.1k

Code Issues Pull requests Actions Projects Wiki Security Insights

main 1 branch 0 tags

Go to file Code

About

Jupyter notebooks for the Natural Language Processing with Transformers book

transformersbook.com/

Readme Apache-2.0 License 1.1k stars 33 watching 170 forks

Releases

No releases published

Packages

File/Folder	Description	Last Commit
.github/ISSUE_TEMPLATE	Update issue templates	25 days ago
data	Move dataset to data directory	4 months ago
images	Add README	last month
scripts	Update issue templates	25 days ago
.gitignore	Initial commit	4 months ago
01_introduction.ipynb	Remove Colab badges & fastdoc refs	27 days ago
02_classification.ipynb	Merge pull request #8 from nlp-with-transformers/remove-display-df	26 days ago
03_transformer-anatomy.ipynb	[Transformers Anatomy] Remove cells with figure references	22 days ago
04_multilingual-ner.ipynb	Merge pull request #8 from nlp-with-transformers/remove-display-df	26 days ago
05_text-generation.ipynb	Merge pull request #8 from nlp-with-transformers/remove-display-df	26 days ago

lewtun Merge pull request #21 from JingchaoZhang/patch-3 ... ae5b7c1 15 days ago 71 commits

O'REILLY

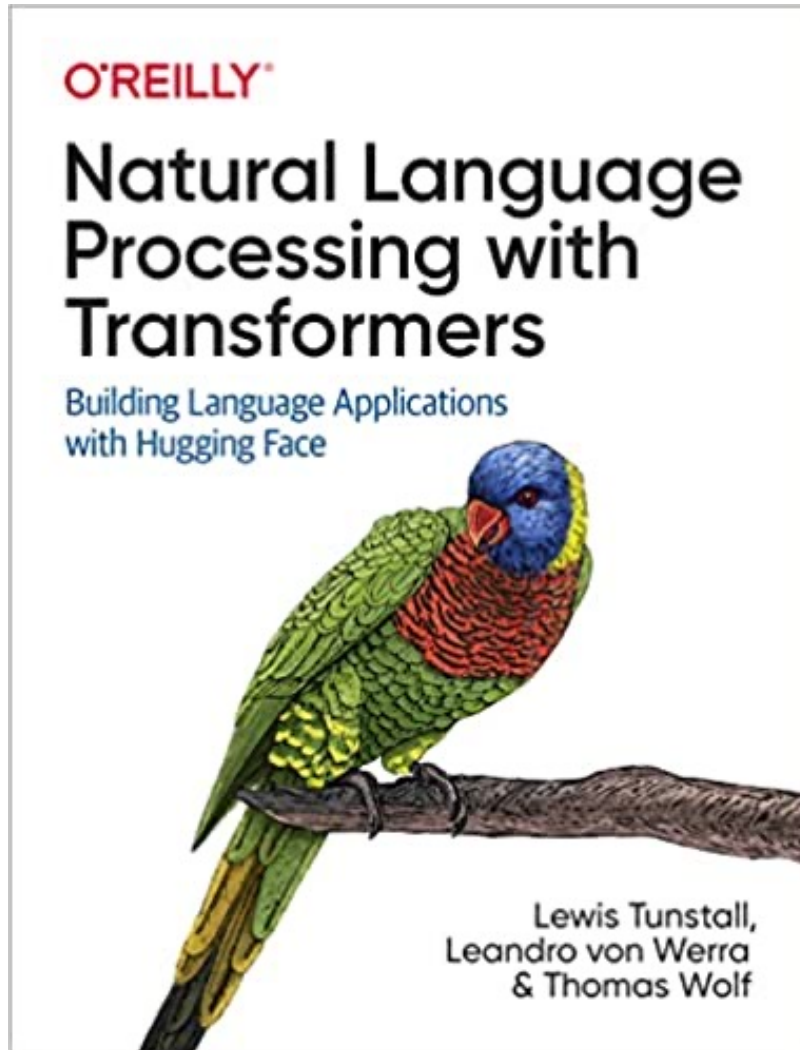
Natural Language Processing with Transformers

Building Language Applications with Hugging Face

Lewis Tunstall, Leandro van Werra & Thomas Wolf

<https://github.com/nlp-with-transformers/notebooks>

NLP with Transformers Github Notebooks



Running on a cloud platform

To run these notebooks on a cloud platform, just click on one of the badges in the table below:

Chapter	Colab	Kaggle	Gradient	Studio Lab
Introduction	Open in Colab	Open in Kaggle	Run on Gradient	Open Studio Lab
Text Classification	Open in Colab	Open in Kaggle	Run on Gradient	Open Studio Lab
Transformer Anatomy	Open in Colab	Open in Kaggle	Run on Gradient	Open Studio Lab
Multilingual Named Entity Recognition	Open in Colab	Open in Kaggle	Run on Gradient	Open Studio Lab
Text Generation	Open in Colab	Open in Kaggle	Run on Gradient	Open Studio Lab
Summarization	Open in Colab	Open in Kaggle	Run on Gradient	Open Studio Lab
Question Answering	Open in Colab	Open in Kaggle	Run on Gradient	Open Studio Lab
Making Transformers Efficient in Production	Open in Colab	Open in Kaggle	Run on Gradient	Open Studio Lab
Dealing with Few to No Labels	Open in Colab	Open in Kaggle	Run on Gradient	Open Studio Lab
Training Transformers from Scratch	Open in Colab	Open in Kaggle	Run on Gradient	Open Studio Lab
Future Directions	Open in Colab	Open in Kaggle	Run on Gradient	Open Studio Lab

Nowadays, the GPUs on Colab tend to be K80s (which have limited memory), so we recommend using [Kaggle](#), [Gradient](#), or [SageMaker Studio Lab](#). These platforms tend to provide more performant GPUs like P100s, all for free!

<https://github.com/nlp-with-transformers/notebooks>

Python in Google Colab (Python101)

<https://colab.research.google.com/drive/1FEG6DnGvwfUbeo4zJ1zTunjMqf2RkCrT>

The screenshot shows a Google Colab notebook interface. At the top, the title is 'python101.ipynb' with a star icon. Below the title, there are menu options: File, Edit, View, Insert, Runtime, Tools, Help, and 'All changes saved'. The main heading is 'NLP with Transformers' in large red font. On the right side, there are icons for Comment, Share, Settings, and a user profile. Below these are RAM and Disk usage indicators and an 'Editing' mode toggle.

On the left, there is a 'Table of contents' sidebar with a search icon. The contents are organized into sections: 'Natural Language Processing with Transformers' (expanded), 'AI in Finance', and 'Data Driven Finance'. Under 'Natural Language Processing with Transformers', there are sub-items: Text Classification, Named Entity Recognition, Question Answering, Summarization, Translation, and Text Generation. Under 'AI in Finance', there are 'Normative Finance and Financial Theories' (expanded) and 'Uncertainty and Risk'. Under 'Normative Finance and Financial Theories', there are 'Expected Utility Theory (EUT)', 'Mean-Variance Portfolio Theory (MVPT)', 'Capital Asset Pricing Model (CAPM)', and 'Arbitrage Pricing Theory (APT)'. Under 'Data Driven Finance', there are 'Financial Econometrics and Regression' and 'Data Availability'. Under 'Normative Theories Revisited', there is 'Mean-Variance Portfolio Theory'.

The main content area shows the following code cells:

- Natural Language Processing with Transformers**
 - Source: Lewis Tunstall, Leandro von Werra, and Thomas Wolf (2022), Natural Language Processing with Transformers: Building Language Applications with Hugging Face, O'Reilly Media.
 - Github: <https://github.com/nlp-with-transformers/notebooks>
- [1]

```
1 !git clone https://github.com/nlp-with-transformers/notebooks.git
2 %cd notebooks
3 from install import *
4 install_requirements()
```
- [3]

```
1 from utils import *
2 setup_chapter()
```
- [12]

```
1 text = """Dear Amazon, last week I ordered an Optimus Prime action figure \
2 from your online store in Germany. Unfortunately, when I opened the package, \
3 I discovered to my horror that I had been sent an action figure of Megatron \
4 instead! As a lifelong enemy of the Decepticons, I hope you can understand my \
5 dilemma. To resolve the issue, I demand an exchange of Megatron for the \
6 Optimus Prime figure I ordered. Enclosed are copies of my records concerning \
7 this purchase. I expect to hear from you soon. Sincerely, Bumblebee."""
```
- Text Classification**
- [13]

```
1 from transformers import pipeline
2 classifier = pipeline("text-classification")
```
- [14]

```
1 import pandas as pd
2 outputs = classifier(text)
3 pd.DataFrame(outputs)
```

<https://tinyurl.com/aintpupython101>

Python in Google Colab (Python101)

<https://colab.research.google.com/drive/1FEG6DnGvwfUbeo4zJ1zTunjMqf2RkCrT>

The screenshot shows a Google Colab notebook interface. At the top, the title 'Text Classification' is displayed in large red font. Below the title, the notebook name 'python101.ipynb' and a star icon are visible. The top navigation bar includes 'File', 'Edit', 'View', 'Insert', 'Runtime', 'Tools', and 'Help', with a status indicator 'All changes saved'. On the right side, there are icons for 'Comment', 'Share', and a user profile 'A'. Below the navigation bar, there are RAM and Disk usage indicators, and a status 'Editing'. The main content area is divided into a left sidebar and a main workspace. The sidebar, titled 'Table of contents', lists the following sections: 'Text Classification with Transformers' (expanded), 'The Dataset', 'From Datasets to DataFrames', 'From Text to Tokens' (with sub-items: 'Character Tokenization', 'Word Tokenization', 'Subword Tokenization'), 'Tokenizing the Whole Dataset', 'Training a Text Classifier' (with sub-items: 'Transformers as Feature Extractors', 'Extracting the last hidden states', 'Creating a feature matrix', 'Visualizing the training set', 'Training a simple classifier'), 'Fine-Tuning Transformers' (with sub-items: 'Loading a pretrained model', 'Defining the performance metrics', 'Training the model'), 'Sidebar: Fine-Tuning with Keras', 'Error analysis', and 'Saving and sharing the model'. The main workspace shows the following content: a section header 'Text Classification with Transformers' with a list of bullet points: 'Source: Lewis Tunstall, Leandro von Werra, and Thomas Wolf (2022), Natural Language Processing with Transformers: Building Language Applications with Hugging Face, O'Reilly Media.' and 'Github: <https://github.com/nlp-with-transformers/notebooks>'. Below this, there are three code cells. The first cell, labeled '[10]', contains the command `!nvidia-smi`. The second cell, labeled '[11]', contains a comment `# Uncomment and run this cell if you're on Colab or Kaggle` followed by the commands `!git clone https://github.com/nlp-with-transformers/notebooks.git`, `%cd notebooks`, `from install import *`, and `install_requirements()`. The third cell, labeled '[12]', contains the commands `# hide`, `from utils import *`, and `setup_chapter()`. Below the code cells, there is a section header 'The Dataset' followed by a code cell labeled '[13]' containing the commands `from datasets import list_datasets`, `all_datasets = list_datasets()`, `print(f"There are {len(all_datasets)} datasets currently available on the Hub")`, and `print(f"The first 10 are: {all_datasets[:10]}")`. The output of this cell shows: 'There are 3783 datasets currently available on the Hub' and 'The first 10 are: ['acronym_identification', 'ade_corpus_v2', 'adversarial_ga', ...]'. At the bottom of the notebook, there is a URL <https://tinyurl.com/aintpupython101>.

<https://tinyurl.com/aintpupython101>

Python in Google Colab (Python101)

<https://colab.research.google.com/drive/1FEG6DnGvwfUbeo4zJ1zTunjMqf2RkCrT>



python101.ipynb ☆

Named Entity Recognition (NER)

File Edit View Insert Runtime Tools Help All changes saved

Comment

Share



A

+ Code + Text

RAM
Disk

Editing



▾ Multilingual Named Entity Recognition (NER)

- Source: Lewis Tunstall, Leandro von Werra, and Thomas Wolf (2022), Natural Language Processing with Transformers: Building Language Applications with Hugging Face, O'Reilly Media.
- Github: <https://github.com/nlp-with-transformers/notebooks>

```
[ ] 1 #NER: https://huggingface.co/tasks/token-classification
2 !pip install transformers
3 from transformers import pipeline
4 classifier = pipeline("ner")
5 classifier("Hello I'm Omar and I live in Zürich.")
```

```
▶ 1 from transformers import pipeline
2 classifier = pipeline("ner")
3 classifier("Hello I'm Omar and I live in Zürich.")
```

```
↳ No model was supplied, defaulted to dbmdz/bert-large-cased-finetuned-conll103-english (https://huggingface.co/dbmdz/bert-large-cased-finetuned-conll103-eng)
[{'end': 14,
  'entity': 'I-PER',
  'index': 5,
  'score': 0.99770516,
  'start': 10,
  'word': 'Omar'},
 {'end': 35,
  'entity': 'I-LOC',
  'index': 10,
  'score': 0.9968976,
  'start': 29,
  'word': 'Zürich'}]
```

<https://tinyurl.com/aintpupython101>

Python in Google Colab (Python101)

<https://colab.research.google.com/drive/1FEG6DnGvwfUbeo4zJ1zTunjMqf2RkCrT>

python101.ipynb ☆

File Edit View Insert Runtime Tools Help Saving...

Text Summarization

- Source: Lewis Tunstall, Leandro von Werra, and Thomas Wolf (2022), Natural Language Processing with Transformers: Building Language Applications with Hugging Face, O'Reilly Media.
- Github: <https://github.com/nlp-with-transformers/notebooks>

```
1 #Source: https://huggingface.co/tasks/summarization
2 !pip install transformers
3 from transformers import pipeline
4 classifier = pipeline("summarization")
5 text = "Paris is the capital and most populous city of France, with an estimated population of 2,175,601 residents as of 2018, in an area of more than 105 km². The city is the most populous in France and the European Union."
6 classifier(text, max_length=30)
```

No model was supplied, defaulted to sshleifer/distilbart-cnn-12-6 (<https://huggingface.co/sshleifer/distilbart-cnn-12-6>)
Your min_length=56 must be inferior than your max_length=30.
[{'summary_text': ' Paris is the capital and most populous city of France, with an estimated population of 2,175,601 residents . The City of Paris'}]

```
1 #!pip install transformers
2 text = """Dear Amazon, last week I ordered an Optimus Prime action figure \
3 from your online store in Germany. Unfortunately, when I opened the package, \
4 I discovered to my horror that I had been sent an action figure of Megatron \
5 instead! As a lifelong enemy of the Decepticons, I hope you can understand my \
6 dilemma. To resolve the issue, I demand an exchange of Megatron for the \
7 Optimus Prime figure I ordered. Enclosed are copies of my records concerning \
8 this purchase. I expect to hear from you soon. Sincerely, Bumblebee."""
9 from transformers import pipeline
10 summarizer = pipeline("summarization")
11 outputs = summarizer(text, max_length=45, clean_up_tokenization_spaces=True)
12 print(outputs[0]['summary_text'])
```

<https://tinyurl.com/aintpupython101>

Python in Google Colab (Python101)

<https://colab.research.google.com/drive/1FEG6DnGvwfUbeo4zJ1zTunjMqf2RkCrT>

The screenshot shows a Google Colab notebook interface. At the top, the title is 'python101.ipynb' and the main heading is 'Text Generation' in red. The notebook contains several code cells. The first cell is a text block with source information and a GitHub link. The second cell is a code block that sets up a GPT-2 pipeline and generates three outputs. The third cell is another code block that generates a single output. The fourth cell shows the output of the second code block, which is a list of three generated text snippets. The fifth cell shows the output of the third code block, which is a single generated text snippet.

python101.ipynb ☆

File Edit View Insert Runtime Tools Help All changes saved

Text Generation

- Source: Lewis Tunstall, Leandro von Werra, and Thomas Wolf (2022), Natural Language Processing with Transformers: Building Language Applications with Hugging Face, O'Reilly Media.
- Github: <https://github.com/nlp-with-transformers/notebooks>

```
[9] 1 #Source: https://huggingface.co/tasks/text-generation
2 #!pip install transformers
3 from transformers import pipeline
4 generator = pipeline('text-generation', model = 'gpt2')
5 generator("Hello, I'm a language model", max_length = 30, num_return_sequences=3)
```

Setting `pad_token_id` to `eos_token_id`:50256 for open-end generation.

```
[{'generated_text': "Hello, I'm a language model.\n\nBut then, one day, I'm not trying to teach the language in my head.\n\n"},
{'generated_text': "Hello, I'm a language model. I'm an implementation for the type system. I'm working with types and programming language constructs. I a
{'generated_text': "Hello, I'm a language modeler, not a programmer. As you know, languages are not a linear model. The thing that jumps out at"}]
```

```
1 from transformers import pipeline
2 generator = pipeline('text-generation', model = 'gpt2')
3 outputs = generator("Once upon a time", max_length = 30, num_return_sequences=3)
4 print(outputs[0]['generated_text'])
```

Setting `pad_token_id` to `eos_token_id`:50256 for open-end generation.

```
Once upon a time, every person who ever saw Jesus, knew that He was Christ. And even though he might not have known Him, He was
```

```
[1] 1 from transformers import pipeline
```

<https://tinyurl.com/aintpupython101>

Python in Google Colab (Python101)

<https://colab.research.google.com/drive/1FEG6DnGvwfUbeo4zJ1zTunjMqf2RkCrT>

The screenshot shows a Google Colab notebook interface. At the top, the notebook title is 'python101.ipynb'. Below the title bar, there are navigation icons for 'Code' and 'Text', and a large red title 'Question Answering and Dialogue Systems' is overlaid on the page. The notebook content is organized into sections: 'Question Answering and Dialogue Systems' and 'Question Answering'. Under 'Question Answering', there is a code cell with the following Python code:

```
[3] 1 !pip install transformers
2 from transformers import pipeline
3 qamodel = pipeline("question-answering")
4 question = "Where do I live?"
5 context = "My name is Michael and I live in Taipei."
6 qamodel(question = question, context = context)
```

Below the code cell, the execution output is shown, including a warning message and the final JSON result:

```
1 from transformers import pipeline
2 qamodel = pipeline("question-answering")
3 question = "Where do I live?"
4 context = "My name is Michael and I live in Taipei."
5 qamodel(question = question, context = context)
6 #{'answer': 'Taipei', 'end': 39, 'score': 0.9730741381645203, 'start': 33}
```

No model was supplied, defaulted to distilbert-base-cased-distilled-squad (<https://huggingface.co/distilbert-base-cased-distilled-squad>)
{'answer': 'Taipei', 'end': 39, 'score': 0.9730741381645203, 'start': 33}

<https://tinyurl.com/aintpupython101>

Python in Google Colab (Python101)

<https://colab.research.google.com/drive/1FEG6DnGvwfUbeo4zJ1zTunjMqf2RkCrT>

python101.ipynb ☆

File Edit View Insert Runtime Tools Help All changes saved

Comment Share ⚙️ A

+ Code + Text

Question Answering

```
[12] 1 from transformers import pipeline
      2 qamodel = pipeline("question-answering", model='deepset/roberta-base-squad2')
      3 question = "What causes precipitation to fall?"
      4 context = """In meteorology, precipitation is any product of the condensation of atmospheric water vapor that falls under gravi·
      5 output = qamodel(question = question, context = context)
      6 print(output['answer'])
```

gravity

```
[13] 1 from transformers import pipeline
      2 qamodel = pipeline("question-answering", model='deepset/roberta-base-squad2')
      3 question = "What is another main form of precipitation besides drizzle, rain, snow, sleet and hail?"
      4 context = """In meteorology, precipitation is any product of the condensation of atmospheric water vapor that falls under gravi·
      5 output = qamodel(question = question, context = context)
      6 print(output['answer'])
```

graupel

```
1 #from transformers import pipeline
2 #qamodel = pipeline("question-answering", model='deepset/roberta-base-squad2')
3 question = "Where do water droplets collide with ice crystals to form precipitation?"
4 context = """In meteorology, precipitation is any product of the condensation of atmospheric water vapor that falls under gravi·
5 output = qamodel(question = question, context = context)
6 print(output['answer'])
```

within a cloud

<https://tinyurl.com/aintpupython101>

Python in Google Colab (Python101)

<https://colab.research.google.com/drive/1FEG6DnGvwfUbeo4zJ1zTunjMqf2RkCrT>

python101.ipynb ☆
File Edit View Insert Runtime Tools Help All changes saved

RAM Disk Editing

Question Answering and Dialogue Systems

Question Answering (QA)

BERT for Question Answering

Source: Apoorv Nandan (2020), BERT (from HuggingFace Transformers) for Text Extraction, https://keras.io/examples/nlp/text_extraction_with_bert/

Description: Fine tune pretrained BERT from HuggingFace Transformers on SQuAD.

Introduction

This demonstration uses SQuAD (Stanford Question-Answering Dataset). In SQuAD, an input consists of a question, and a paragraph for context. The goal is to find the span of text in the paragraph that answers the question. We evaluate our performance on this data with the "Exact Match" metric, which measures the percentage of predictions that exactly match any one of the ground-truth answers.

We fine-tune a BERT model to perform this task as follows:

1. Feed the context and the question as inputs to BERT.
2. Take two vectors S and T with dimensions equal to that of hidden states in BERT.
3. Compute the probability of each token being the start and end of the answer span. The probability of a token being the start of the answer is given by a dot product between S and the representation of the token in the last layer of BERT, followed by a softmax over all tokens. The probability of a token being the end of the answer is computed similarly with the vector T .
4. Fine-tune BERT and learn S and T along the way.

References:

- [BERT](#)
- [SQuAD](#)

<https://tinyurl.com/aintpupython101>

Python in Google Colab (Python101)

<https://colab.research.google.com/drive/1FEG6DnGvwfUbeo4zJ1zTunjMqf2RkCrT>

python101.ipynb ☆

File Edit View Insert Runtime Tools Help All changes saved

Comment Share Settings A

RAM Disk Editing

Table of contents

- RandomizedSearchCV
- Sentiment Analysis
 - Sentiment Analysis - Unsupervised Lexical
 - Sentiment Analysis - Supervised Machine Learning
 - Sentiment Analysis - Supervised Deep Learning Models
 - Sentiment Analysis - Advanced Deep Learning
- Deep Learning and Universal Sentence-Embedding Models
 - Universal Sentence Encoder (USE)
 - Universal Sentence Encoder Multilingual (USEM)
- Question Answering and Dialogue Systems
 - Question Answering (QA)
 - BERT for Question Answering**
 - Dialogue Systems
 - Joint Intent Classification and Slot Filling with Transformers
- Data Visualization
- Section

+ Code + Text

```
Downloading: 100% ██████████ 433/433 [00:29<00:00, 14.5B/s]

Downloading: 100% ██████████ 536M/536M [00:29<00:00, 18.3MB/s]

Model: "model"
-----
Layer (type)                Output Shape           Param #   Connected to
-----
input_1 (InputLayer)        [(None, 384)]          0         input_1[0][0]
input_3 (InputLayer)        [(None, 384)]          0         input_3[0][0]
input_2 (InputLayer)        [(None, 384)]          0         input_2[0][0]
tf_bert_model (TFBertModel) ((None, 384, 768), ( 109482240  input_1[0][0]
start_logits (Dense)        (None, 384, 1)         768       tf_bert_model[0][0]
end_logits (Dense)          (None, 384, 1)         768       tf_bert_model[0][0]
flatten (Flatten)           (None, 384)            0         start_logits[0][0]
flatten_1 (Flatten)         (None, 384)            0         end_logits[0][0]
activation_7 (Activation)   (None, 384)            0         flatten[0][0]
activation_8 (Activation)   (None, 384)            0         flatten_1[0][0]
-----
Total params: 109,483,776
Trainable params: 109,483,776
Non-trainable params: 0

CPU times: user 20.8 s, sys: 7.75 s, total: 28.5 s
Wall time: 1min 42s
```

<https://tinyurl.com/aintpupython101>

Python in Google Colab (Python101)

<https://colab.research.google.com/drive/1FEG6DnGvwfUbeo4zJ1zTunjMqf2RkCrT>

The screenshot shows a Google Colab notebook interface. At the top, the notebook is titled "python101.ipynb" and has a star icon. The menu bar includes "File", "Edit", "View", "Insert", "Runtime", "Tools", and "Help", with a status indicator "All changes saved". On the right, there are icons for "Comment", "Share", "Settings", and a user profile icon. Below the menu bar, there are RAM and Disk usage indicators and an "Editing" mode indicator.

The left sidebar shows a "Table of contents" with a search icon and a list of sections: "RandomizedSearchCV", "Sentiment Analysis", "Sentiment Analysis - Unsupervised Lexical", "Sentiment Analysis - Supervised Machine Learning", "Sentiment Analysis - Supervised Deep Learning Models", "Sentiment Analysis - Advanced Deep Learning", "Deep Learning and Universal Sentence-Embedding Models", "Universal Sentence Encoder (USE)", "Universal Sentence Encoder Multilingual (USEM)", "Question Answering and Dialogue Systems", "Question Answering (QA)", "BERT for Question Answering", "Dialogue Systems", "Joint Intent Classification and Slot Filling with Transformers" (highlighted), and "Data Visualization".

The main content area has a "Dialogue Systems" section header in red. Below it is a code cell with the following code:

```
[ ] 1 #Source: Olivier Grisel (2020), Transformers (BERT fine-tuning): Joint Intent Classification and S
     2 #https://github.com/m2dsupsdclass/lectures-labs/blob/master/labs/06_deep_nlp/Transformers_Joint_I
```

Below the code cell is a text cell with the following content:

Joint Intent Classification and Slot Filling with Transformers

The goal of this notebook is to fine-tune a pretrained transformer-based neural network model to convert a user query expressed in English into a representation that is structured enough to be processed by an automated service.

Here is an example of interpretation computed by such a Natural Language Understanding system:

```
>>> nlu("Book a table for two at Le Ritz for Friday night",
        tokenizer, joint_model, intent_names, slot_names)
```

```
{
  'intent': 'BookRestaurant',
  'slots': {
    'party_size_number': 'two',
    'restaurant_name': 'Le Ritz',
    'timeRange': 'Friday night'
  }
}
```

Intent classification is a simple sequence classification problem. The trick is to treat the structured knowledge extraction part ("Slot Filling") as token-level classification problem using BIO-annotations:

<https://tinyurl.com/aintpupython101>

Python in Google Colab (Python101)

<https://colab.research.google.com/drive/1FEG6DnGvwfUbeo4zJ1zTunjMqf2RkCrT>

The screenshot shows a Google Colab notebook titled "python101.ipynb". The interface includes a top menu bar with "File", "Edit", "View", "Insert", "Runtime", "Tools", and "Help", along with a "Comment" button and a "Share" button. A "Table of contents" sidebar on the left lists various topics, with "Joint Intent Classification and Slot Filling with Transformers" highlighted. The main code cell contains a Python function `show_predictions` that takes text, a tokenizer, a model, intent names, and slot names as input. The function uses TensorFlow to process the text and returns the intent and slot names. The output of the function is displayed below the code cell, showing the text "Book a table for two at Le Ritz for Friday night!" and the corresponding intent "BookRestaurant" and slots: "Book", "a", "table", "for", "two", "at", "Le", "R", "##itz", "for", "Friday", "night", and "!".

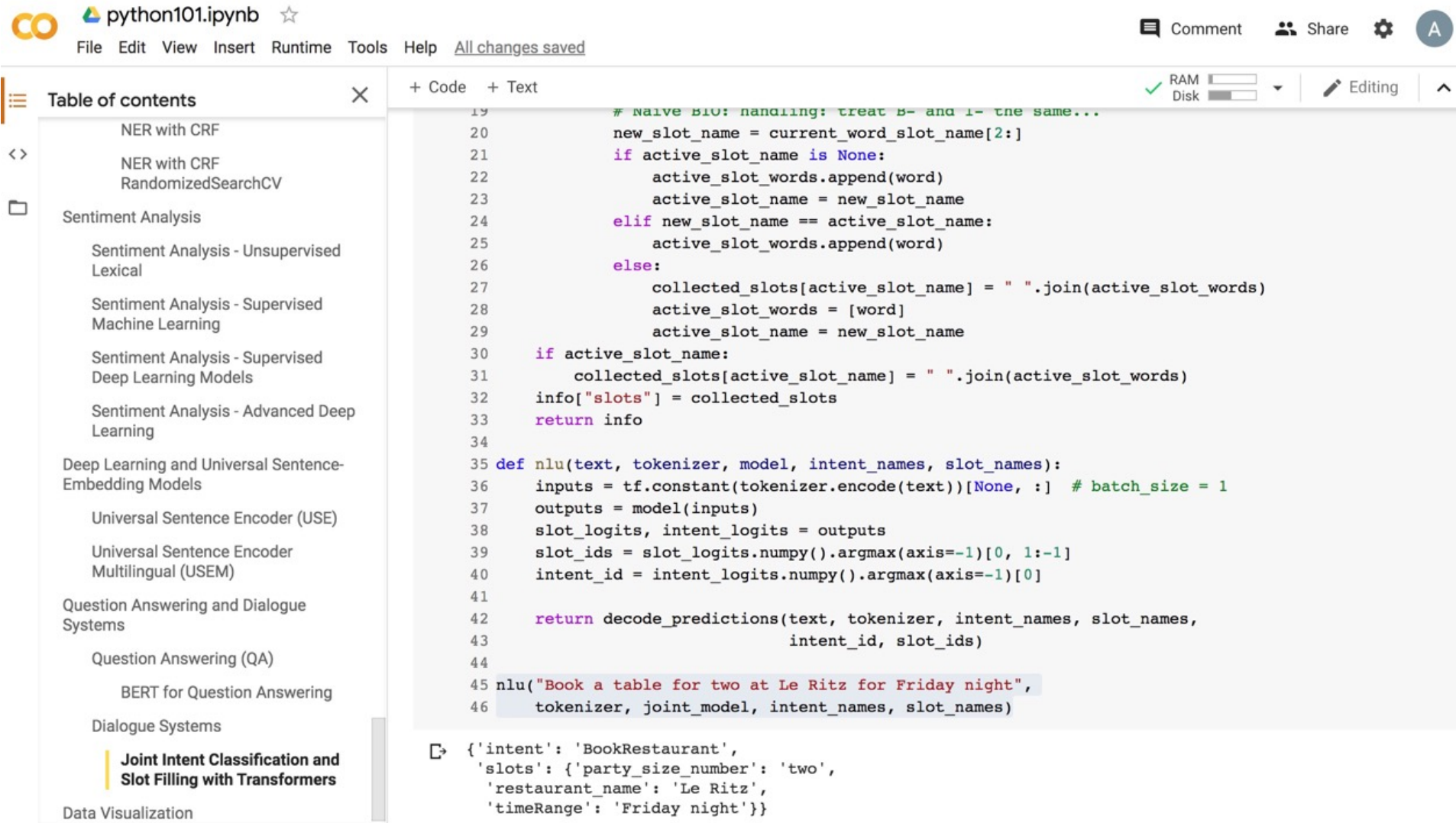
```
1 def show_predictions(text, tokenizer, model, intent_names, slot_names):
2     inputs = tf.constant(tokenizer.encode(text))[None, :] # batch_size = 1
3     outputs = model(inputs)
4     slot_logits, intent_logits = outputs
5     slot_ids = slot_logits.numpy().argmax(axis=-1)[0, 1:-1]
6     intent_id = intent_logits.numpy().argmax(axis=-1)[0]
7     print("Text:", text)
8     print("Intent:", intent_names[intent_id])
9     print("Slots:")
10    for token, slot_id in zip(tokenizer.tokenize(text), slot_ids):
11        print(f"{token:>10} : {slot_names[slot_id]}")
12
13 show_predictions("Book a table for two at Le Ritz for Friday night!",
14                 tokenizer, joint_model, intent_names, slot_names)
```

Text: Book a table for two at Le Ritz for Friday night!
Intent: BookRestaurant
Slots:
Book : 0
a : 0
table : 0
for : 0
two : B-party_size_number
at : 0
Le : B-restaurant_name
R : I-restaurant_name
##itz : I-restaurant_name
for : 0
Friday : B-timeRange
night : 0
! : 0

<https://tinyurl.com/aintpupython101>

Python in Google Colab (Python101)

<https://colab.research.google.com/drive/1FEG6DnGvwfUbeo4zJ1zTunjMqf2RkCrT>



The screenshot displays a Google Colab notebook interface. On the left, a 'Table of contents' sidebar lists various topics, with 'Joint Intent Classification and Slot Filling with Transformers' highlighted. The main area shows a Python script for NER. The script includes a function to process a word and update slot information, and a main function that uses a TensorFlow model to process a sentence. The output of the main function is a dictionary showing the intent and extracted slots.

```
19 # Naive BIOES-style handling: treat B- and I- the same...
20 new_slot_name = current_word_slot_name[2:]
21 if active_slot_name is None:
22     active_slot_words.append(word)
23     active_slot_name = new_slot_name
24 elif new_slot_name == active_slot_name:
25     active_slot_words.append(word)
26 else:
27     collected_slots[active_slot_name] = " ".join(active_slot_words)
28     active_slot_words = [word]
29     active_slot_name = new_slot_name
30 if active_slot_name:
31     collected_slots[active_slot_name] = " ".join(active_slot_words)
32 info["slots"] = collected_slots
33 return info
34
35 def nlu(text, tokenizer, model, intent_names, slot_names):
36     inputs = tf.constant(tokenizer.encode(text))[None, :] # batch_size = 1
37     outputs = model(inputs)
38     slot_logits, intent_logits = outputs
39     slot_ids = slot_logits.numpy().argmax(axis=-1)[0, 1:-1]
40     intent_id = intent_logits.numpy().argmax(axis=-1)[0]
41
42     return decode_predictions(text, tokenizer, intent_names, slot_names,
43                             intent_id, slot_ids)
44
45 nlu("Book a table for two at Le Ritz for Friday night",
46     tokenizer, joint_model, intent_names, slot_names)
```

```
{'intent': 'BookRestaurant',
 'slots': {'party_size_number': 'two',
 'restaurant_name': 'Le Ritz',
 'timeRange': 'Friday night'}}
```

<https://tinyurl.com/aintpupython101>

Question Answering

```
!pip install transformers
from transformers import pipeline
qamodel = pipeline("question-answering")
question = "Where do I live?"
context = "My name is Michael and I live in Taipei."
qamodel(question = question, context = context)
```

```
{'answer': 'Taipei', 'end': 39, 'score': 0.9730741381645203, 'start': 33}
```

Question Answering

```
from transformers import pipeline
qamodel = pipeline("question-answering", model='deepset/roberta-base-squad2')
question = "Where do I live?"
context = "My name is Michael and I live in Taipei."
output = qamodel(question = question, context = context)
print(output[ 'answer' ])
```

Taipei

Question Answering

```
from transformers import pipeline
qamodel = pipeline("question-answering", model='deepset/roberta-base-squad2')
question = "What causes precipitation to fall?"
context = """In meteorology, precipitation is any product of
the condensation of atmospheric water vapor that falls under
gravity. The main forms of precipitation include drizzle,
rain, sleet, snow, graupel and hail... Precipitation forms as
smaller droplets coalesce via collision with other rain drops
or ice crystals within a cloud. Short, intense periods of
rain in scattered locations are called "showers"."""
output = qamodel(question = question, context = context)
print(output['answer'])
```

gravity

Summary

- **Question Answering**
- **Dialogue Systems**
- **Task Oriented Dialogue System**

References

- Lewis Tunstall, Leandro von Werra, and Thomas Wolf (2022), Natural Language Processing with Transformers: Building Language Applications with Hugging Face, O'Reilly Media.
- Denis Rothman (2021), Transformers for Natural Language Processing: Build innovative deep neural network architectures for NLP with Python, PyTorch, TensorFlow, BERT, RoBERTa, and more, Packt Publishing.
- Savaş Yıldırım and Meysam Asgari-Chenaghlu (2021), Mastering Transformers: Build state-of-the-art models from scratch with advanced natural language processing techniques, Packt Publishing.
- Sowmya Vajjala, Bodhisattwa Majumder, Anuj Gupta (2020), Practical Natural Language Processing: A Comprehensive Guide to Building Real-World NLP Systems, O'Reilly Media.
- Dipanjan Sarkar (2019), Text Analytics with Python: A Practitioner's Guide to Natural Language Processing, Second Edition. APress.
- Mutabazi, Emmanuel, Jianjun Ni, Guangyi Tang, and Weidong Cao (2021) "A review on medical textual question answering systems based on deep learning approaches." Applied Sciences 11, no. 12 (2021): 5456.
- Deriu, Jan, Alvaro Rodrigo, Arantxa Otegi, Guillermo Echegoyen, Sophie Rosset, Eneko Agirre, and Mark Cieliebak (2021). "Survey on evaluation methods for dialogue systems." Artificial Intelligence Review 54, no. 1 (2021): 755-810.
- Day, Min-Yuh and Chi-Sheng Hung, "AI Affective Conversational Robot with Hybrid Generative-based and Retrieval-based Dialogue Models", in Proceedings of The 20th IEEE International Conference on Information Reuse and Integration for Data Science (IEEE IRI 2019), Los Angeles, CA, USA, July 30 - August 1, 2019.
- Day, Min-Yuh, Chi-Sheng Hung, Yi-Jun Xie, Jih-Yi Chen, Yu-Ling Kuo and Jian-Ting Lin (2019), "IMTKU Emotional Dialogue System for Short Text Conversation at NTCIR-14 STC-3 (CECG) Task", The 14th NTCIR Conference on Evaluation of Information Access Technologies (NTCIR-14), Tokyo, Japan, June 10-13, 2019.
- Zhou, Hao, Minlie Huang, Tianyang Zhang, Xiaoyan Zhu, and Bing Liu. "Emotional chatting machine: emotional conversation generation with internal and external memory." arXiv preprint arXiv:1704.01074 (2017).
- Yu, Kai, Zijian Zhao, Xueyang Wu, Hongtao Lin, and Xuan Liu. "Rich Short Text Conversation Using Semantic Key Controlled Sequence Generation." IEEE/ACM Transactions on Audio, Speech, and Language Processing (2018)
- Borah, Bhiguraj, Dhrubajyoti Pathak, Priyankoo Sarmah, Bidisha Som, and Sukumar Nandi. "Survey of Textbased Chatbot in Perspective of Recent Technologies." In International Conference on Computational Intelligence, Communications, and Business Analytics, pp. 84-96. Springer, Singapore, 2018.
- Haihong, E., Peiqing Niu, Zhongfu Chen, and Meina Song. "A novel bi-directional interrelated model for joint intent detection and slot filling." In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pp. 5467-5471. 2019.
- Rajpurkar, Pranav, Jian Zhang, Konstantin Lopyrev, and Percy Liang. "Squad: 100,000+ questions for machine comprehension of text." arXiv preprint arXiv:1606.05250 (2016).
- Zhu, Qi, Kaili Huang, Zheng Zhang, Xiaoyan Zhu, and Minlie Huang. "Crosswoz: A large-scale chinese cross-domain task-oriented dialogue dataset." arXiv preprint arXiv:2002.11893 (2020).
- Zeng, Zhaohao, Sosuke Kato, Tetsuya Sakai, and Inho Kang (2020), "Overview of the NTCIR-15 Dialogue Evaluation (DialEval-1) Task", Proceedings of NTCIR-15, 2020.
- Benjamin Bengfort, Rebecca Bilbro, and Tony Ojeda (2018), Applied Text Analysis with Python: Enabling Language-Aware Data Products with Machine Learning, O'Reilly.
- Charu C. Aggarwal (2018), Machine Learning for Text, Springer.
- Gabe Ignatow and Rada F. Mihalcea (2017), An Introduction to Text Mining: Research Design, Data Collection, and Analysis, SAGE Publications.
- Rajesh Arumugam (2018), Hands-On Natural Language Processing with Python: A practical guide to applying deep learning architectures to your NLP applications, Packt.
- Jake VanderPlas (2016), Python Data Science Handbook: Essential Tools for Working with Data, O'Reilly Media.
- Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova (2018). "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding." arXiv preprint arXiv:1810.04805.
- The Super Duper NLP Repo, <https://notebooks.quantumstat.com/>
- Jay Alammar (2018), The Illustrated Transformer, <http://jalammar.github.io/illustrated-transformer/>
- Jay Alammar (2019), A Visual Guide to Using BERT for the First Time, <http://jalammar.github.io/a-visual-guide-to-using-bert-for-the-first-time/>
- NLP with Transformer, <https://github.com/nlp-with-transformers/notebooks>
- Min-Yuh Day (2022), Python 101, <https://tinyurl.com/aintpupython101>