

# Задачи NLP NLG, NLP





# Генерация

# Генерация текста

- Один из основных методов изучения языка в компьютерной лингвистике – построение языковых моделей
- Языковая модель пытается смоделировать распределение вероятностей токенов в естественных текстах, предсказывая вероятности следующего (или пропущенного) токена
- Генерация текстов на естественном языке – естественное применение построенной языковой модели

# Генерация текста

N-gram

принципы не отражают потребностей коммуникации  
п . с том для действующего лица , времени и в  
таких случаях требуется правило другого рода ,  
а не терминальных , а второе зависимое  
местоимение самая изменяется и соответственно  
той мере усилий , которая может быть извлечена  
из универсальных принципов

Данная работа представляет собой исследование на  
материале нескольких периодов. В центре внимания автора  
находится проблема изменения отношения между структурой и  
функциями синтаксических конструкций в языках, в том  
числе на материале русского и китайского. Автор проводит  
анализ семантических и морфологических параметров  
синтаксических конструкций в русском языке.

GPT-2

# Языковые модели

## SLM

**Statistical Language Models:**

- N-грамные языковые модели

## NLM

**Neural Language Models:**

- word2vec
- GloVe
- fasttext

## PLM

**Pre-trained Language Models:**

- ELMo
- BERT
- BART
- GPT-1, 2

## LLM

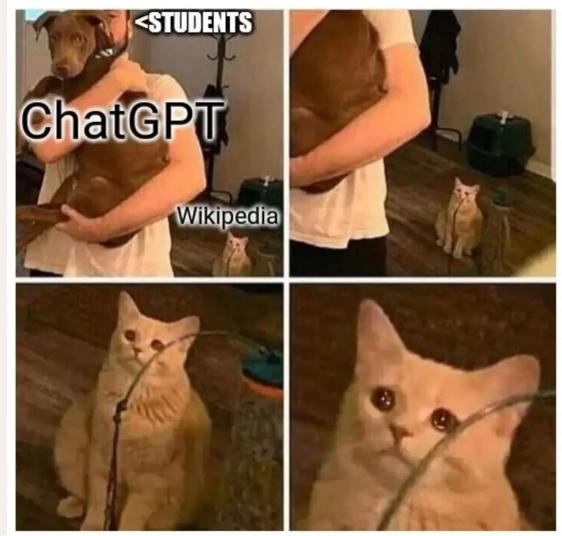
**Large Language Models:**

- GPT-3, 4
- PaLM
- YaLM

# Языковые модели: SOTA

В чем разница между PLM и LLM?

- Очевидно, в размере: LLM – 175 млрд параметров и выше
- LLM показывают новые (удивительные) возможности, каких нет у PLM: они справляются с zero-shot transfer, решают задачи, каких не могут решить PLM (Winograd)
- LLM более универсальные, могут решать несколько задач одновременно, не нуждаясь в специальном дообучении
- Однако LLM – и самые малоизученные языковые модели (помним, они неинтерпретируемые)



# Оценка ЯМ

Уже обсуждали: существует два вида оценки

- Intrinsic evaluation: perplexity
- Extrinsic evaluation: downstream tasks
- Для оценки качества языковых моделей создаются специальные датасеты, предназначенные для разных задач, они называются бенчмарки

# Бенчмарки

- SuperGLUE – для английского языка:
  - 9 разных задач на понимание языка
  - Оценочный датасет с разнообразными языковыми явлениями
  - Публичный лидерборд
- Russian SuperGLUE – для русского языка:

Linguistic Diagnostic for Russian	NLI & diagnostics	LiDiRus
Russian Commitment Bank	NLI	RCB
Choice of Plausible Alternatives for Russian language	Common Sense	PARus
Russian Multi-Sentence Reading Comprehension	Machine Reading	MuSeRC
Textual Entailment Recognition for Russian	NLI	TERRa
Russian Words in Context (based on RUSSE)	Common Sense	RUSSE
The Winograd Schema Challenge (Russian)	Reasoning	RWSD
Yes/no Question Answering Dataset for the Russian	World Knowledge	DaNetQA
Russian Reading Comprehension with Commonsense Reasoning	Machine Reading	RuCoS

# Лидерборды

SuperGlue

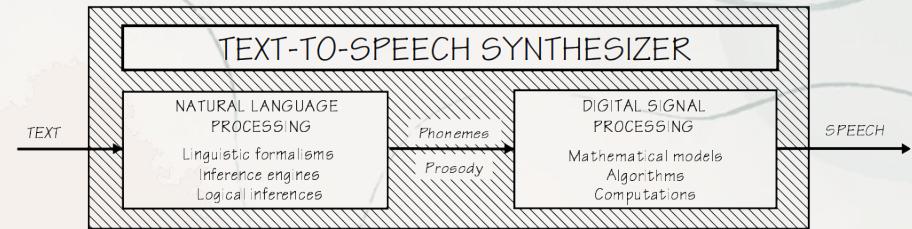
1	Microsoft Alexander v-team	Turing ULR v6
2	JDExplore d-team	Vega v1
3	Microsoft Alexander v-team	Turing NLR v5
4	DIRL Team	DeBERTa + CLEVER
5	ERNIE Team - Baidu	ERNIE
6	AliceMind & DIRL	StructBERT + CLEVER
7	DeBERTa Team - Microsoft	DeBERTa / TuringNLRv4
8	HFL iFLYTEK	MacALBERT + DKM
9	PING-AN Omni-Sinicic	ALBERT + DAAF + NAS
10	T5 Team - Google	T5

Russian  
SuperGlue

Rank	Name	Team
1	HUMAN BENCHMARK	AGI NLP
2	Mistral 7B LoRA	Saiga team
3	FRED-T5 1.7B finetune	SberDevices
4	Golden Transformer v2.0	Avengers Ensemble
5	LLaMA-2 13B LoRA	Saiga team
6	Saiga 13B LoRA	Saiga team
7	YaLM p-tune (3.3B frozen + 40k trainable params)	Yandex
8	FRED-T5 large finetune	SberDevices
9	RuLeanALBERT	Yandex Research
10	FRED-T5 1.7B (only encoder 760M) finetune	SberDevices

# Text-to-Speech

- Обзор 1997 года: «automatic production of speech, through a grapheme-to-phoneme transcription of the sentences to utter»
- Картинка оттуда же
- Донейронная эпоха:
- Letter-to-Sound модуль:
  - на словаре (морфем) vs на правилах
- Уже тогда правила проигрывали даже словарям, но считались перспективными среди лингвистов



# Text-to-Speech

- Современные решения, разумеется, работают на нейронных сетях
- Популярные архитектуры – CNN, RNN, GAN, Diffusion
- Diffusion – SOTA архитектура
- Отлично генерирует не только звук, но и картинки!
- Тут можно музыку генерить
- Раньше использовали трехступенчатый подход: из текста извлекаются языковые признаки, потом акустические, потом синтезируется звук
- Сегодня языковые признаки не используют, только акустические

Table 1. Recent progress of text-to-speech diffusion models

Stage	Category	Methods
Acoustic model	Pioneering work	Diff-TTS [31] Grad-TTS [85]
	Efficient acoustic model	ProDiff [28] DiffGAN-TTS [60]
	Adaptive multi-speaker model	Grad-TTS with ILVR [52] Grad-StyleSpeech [32] Guided-TTS [37] Guided-TTS 2 [38]
	With discrete latent space	Diffsound [128] NoreSpeech [127]
Vocoder	Fine-grained control	EmoDiff [22]
	Pioneering work	WaveGrad [7] DiffWave [45]
	Efficient vocoder	BDDM [48] InferGrad [9] WaveFit [43]
	Statistical improvement	DDGM [70] PriorGrad [50] ItôWave [125] SpecGrad [44]
End-to-end	Pioneering work	WaveGrad 2 [8] CRASH [90]
	Efficient model	FastDiff [26]
	Further improvements	DAG [79] Itôn [99]

# Text-to-Speech

Составляющие сегодняшнего TTS пайплайна:

- Акустическая модель – извлекает акустические признаки из текста (для этого нужны аудиозаписи с транскрипцией)
- Вокодер – генерирует аудиосигнал на основе акустических признаков (это тоже нейронная сеть)
- End-to-end модель совмещает акустическую модель и вокодер: получает текст, должна генерировать звук
- Очень простая python-библиотека - [gTTS](#)

# Text-to-Image

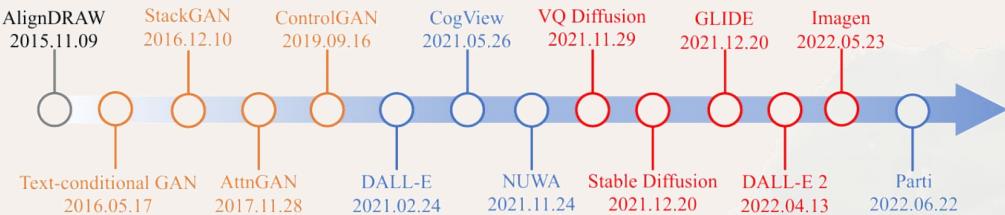
- Сегодняшняя SOTA-архитектура – тоже Diffusion
- Нужна языковая модель – чтобы обработать промпт
- Вектор, полученный из промпта, подмешивается в шум при генерации картинки



# Text-to-Image

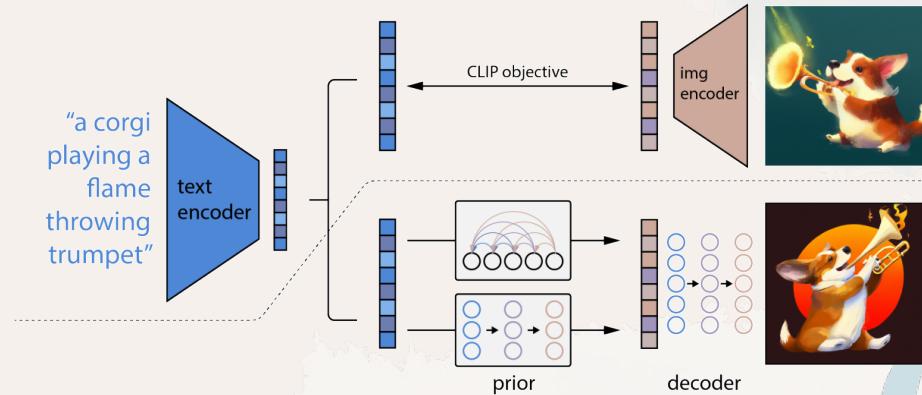
Самые известные модели:

- DALL-E
- Midjourney (проприетарная)
- StableDiffusion
- Kandinsky



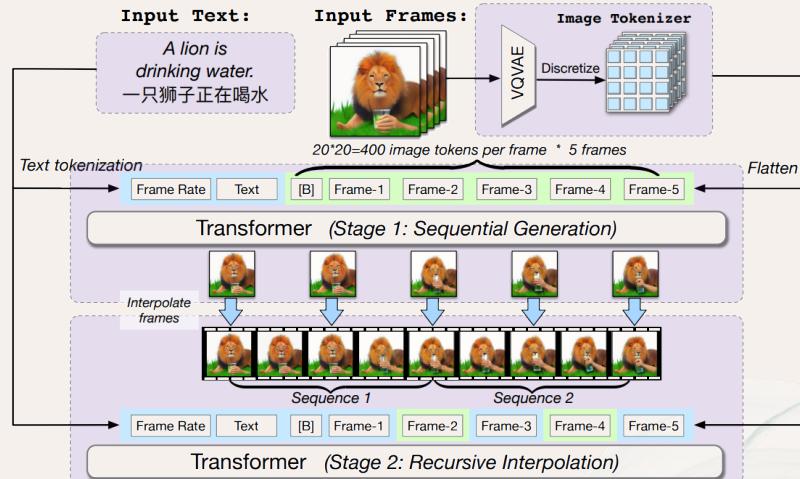
# Text-to-Image

- Используют CLIP (Contrastive Language Image Pre-training): это модель, которая совмещает эмбеддинги для картинки и текста в одном пространстве

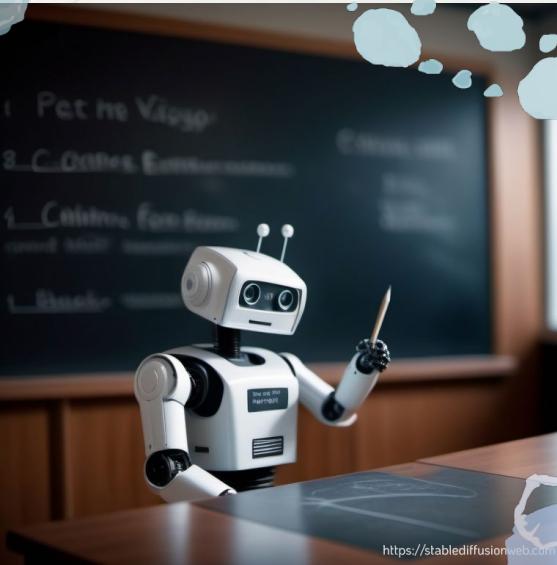


# Text-to-Video

- Задача была поставлена позднее предыдущих
- Активно разрабатываются модели генерации видео
- Еще пока мало датасетов для обучения
- Но уже есть модели: например, [CoqVideo \(2022\)](#) (ссылка на статью, а не потыкать)



# NLU+NLG



<https://stablediffusionweb.com>

# Симплификация

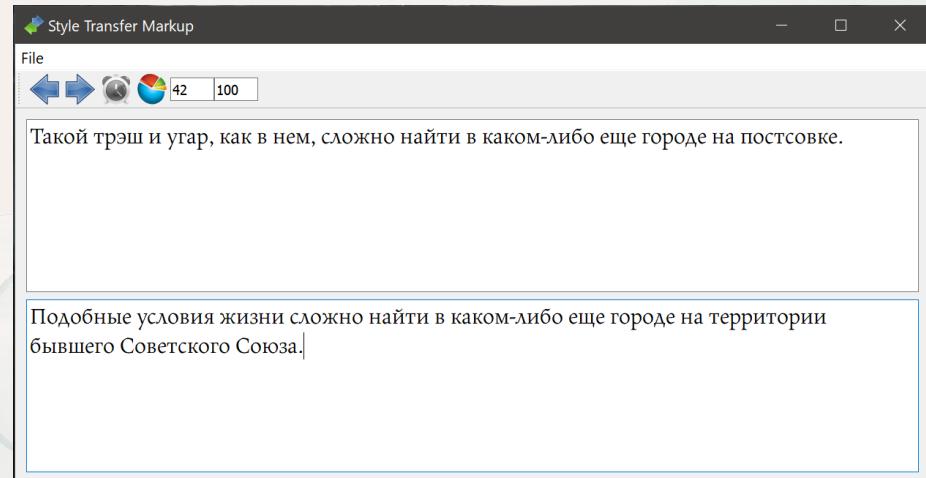
- Нужно: для автоматического упрощения текста при изучении языка, при нарушениях восприятия
- Решает в основном две задачи: упрощение лексики и упрощение синтаксиса
- Не только генерация, но и понимание
- Тесно связано с автоматической оценкой сложности текста
- Про автоматическую оценку сложности целые диссертации пишут
- Существует несколько различных метрик для автоматической оценки сложности: Flesch Reading Ease score, Fog Index, SMOG, Flesch-Kincaid Grade Level index, Bilingual evaluation understudy (BLEU) – последняя также очень важна для оценки МП
- Русскоязычная модель T5

# Парафразирование

- Парафразы – предложения, которые передают одно и то же разными способами
- Тоже тесно связано с NLU
- Используемые метрики – тоже BLEU, ROUGE, METEOR, TER
- Подходы:
  - Правиловые
  - На тезаурусах
  - Статистические (МП)
  - На нейронных сетях
- Много сложностей: например, с сохранением стилистических особенностей
- SOTA далеки от удовлетворительного (самая крутая оценка не дотягивает до 70%)

# Перенос стиля

- Text Style Transfer – переписываем тот же текст (содержательно) в другом функциональном стиле
- Не очень понятно, как определить стиль – математически точных определений нет
- В NLP используют data-driven definition of style, чтобы не заморачиваться с теорией
- Стиль – это набор атрибутов, изменяющихся в разных датасетах (в противовес неизменным атрибутам)



# Перенос стиля

- Одна из подзадач – детоксикация
- Много проблем с определением того, что же нужно «детоксифицировать»
- В 2022 году проходило соревнование RUSSE-2022 Detoxification
- Вопросы к датасету и принципам его создания были даже у участников

Does this text contain offenses or swear words?

I don't care about that.

Yes     No

Do these sentences mean the same?

I don't f\*ckin care about that shit

I don't care about that

Yes     No

Is this text grammatical?

I don't care about that.

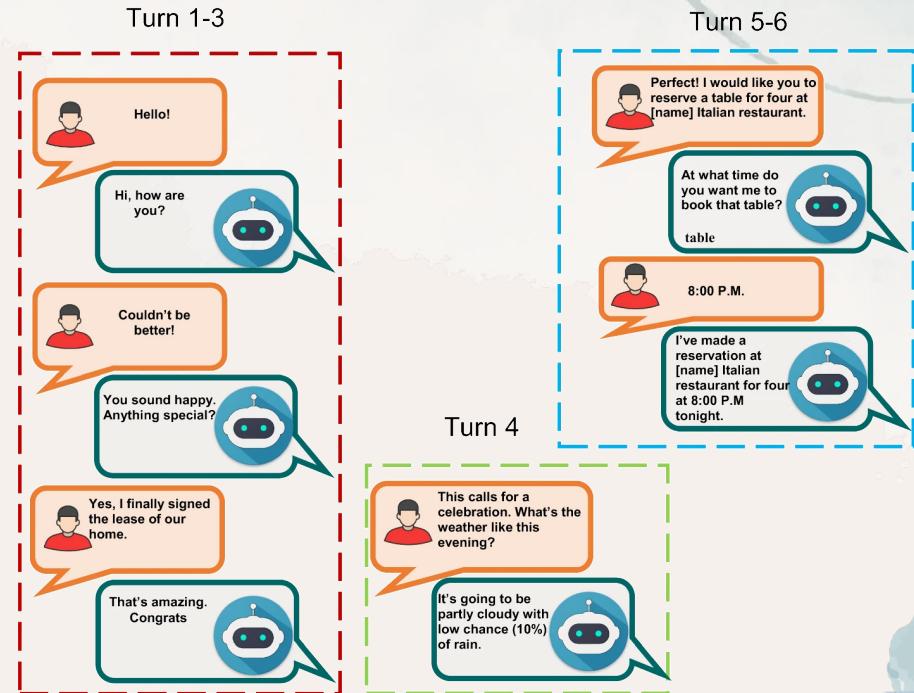
YES, there are no or only minor mistakes

PARTIALLY, there are mistakes, but the text is intelligible

NO, the text is difficult to understand

# Вопросно-ответные системы

- Бывает как минимум двух видов:
  - Knowledge Base Question Answering (KBQA)
  - Conversational Question Answering (CQA)
- KBQA может решаться с помощью методов семантического парсинга или извлечения информации
- CQA – область создания диалоговых систем



# Чатботы и диалоговые системы

- Одна из самых популярных областей NLP
- Два типа: task-oriented vs open-domain dialogue systems
- Ранние подходы: правиловые и на классических алгоритмах МО
- Правиловые очень легко делать! Самый ранний чатбот – ELIZA
- Его написали в 60-х гг.
- Сегодня, конечно, все на нейронных сетях

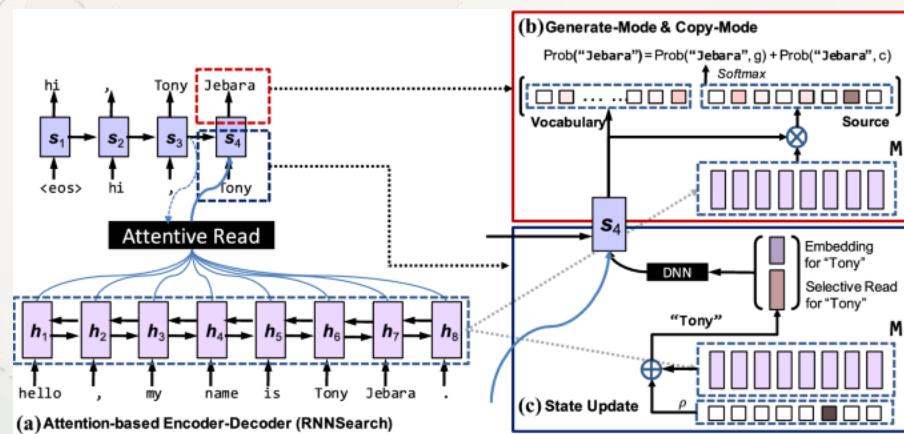
```
Welcome to
      EEEEEE  LL     IIII   ZZZZZZ  AAAAA
      EE     LL     II     ZZ    AA   AA
      EEEEEE  LL     II     ZZZ   AAAAAAA
      EE     LL     II     ZZ    AA   AA
      EEEEEE  LLLLLL  IIII   ZZZZZZ  AA   AA

Eliza is a mock Rogerian psychotherapist.
The original program was described by Joseph Weizenbaum in 1966.
This implementation by Norbert Landsteiner 2005.

ELIZA: Is something troubling you ?
YOU: Men are all alike.
ELIZA: What is the connection, do you suppose ?
YOU: They're always bugging us about something or other.
ELIZA: Can you think of a specific example ?
YOU: Well, my boyfriend made me come here.
ELIZA: Is it important to you that your boyfriend made you come here ?
YOU: He says I'm depressed much of the time.
ELIZA: I am sorry to hear that you are depressed.
YOU: It's true. I am unhappy.
ELIZA: Can you explain what made you unhappy ?
YOU:
```

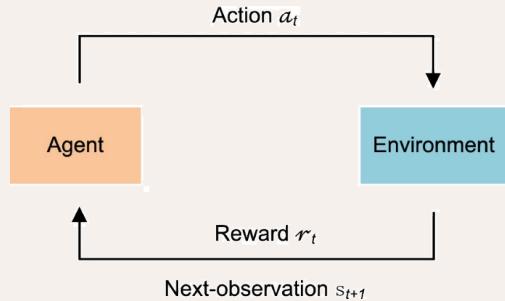
# Чатботы и диалоговые системы

- В дополнение к существующим архитектурам нейронных сетей чатботам нужно уметь не только генерировать, но и копировать некоторые токены из ответов пользователей
- Для этого разрабатываются свои архитектуры (CopyNet)



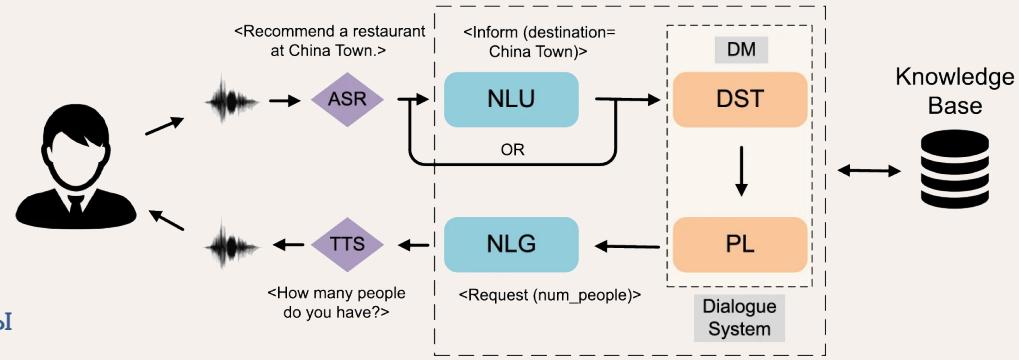
# Чатботы и диалоговые системы

- Не очень понятно, как оценивать диалоговую систему, особенно с открытым доменом: методы обучения с учителем плохо подходят
- Поэтому используется обучение с подкреплением (reinforcement learning)
- Еще возможно использовать GAN
- Иногда диалоговые системы снабжают графиками знаний



# Task-oriented DS

- Пайпайн содержит следующие компоненты:
  - Natural Language Understanding
  - Dialogue State Tracking
  - Policy Learning
  - Natural Language Generation
- Либо создаются End-to-End системы



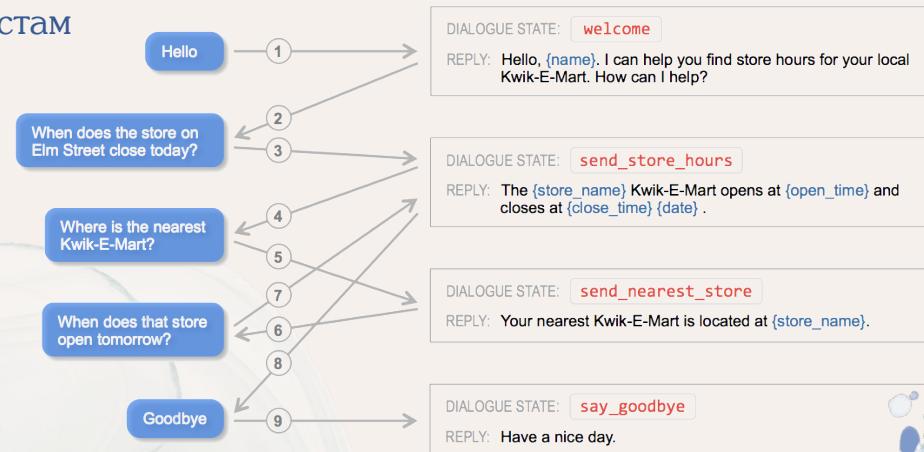
# Task-oriented DS

- Natural Language Understanding включает:
  - Domain Classification
  - Intent Detection
  - Slot Filling
- Иногда эти три задачи объединяют в одну архитектуру (Multi-task)
- Также используется модуль для распознания речи (но есть подходы, когда в NLU-юнит передается сразу звуковой сигнал)

# Task-oriented DS

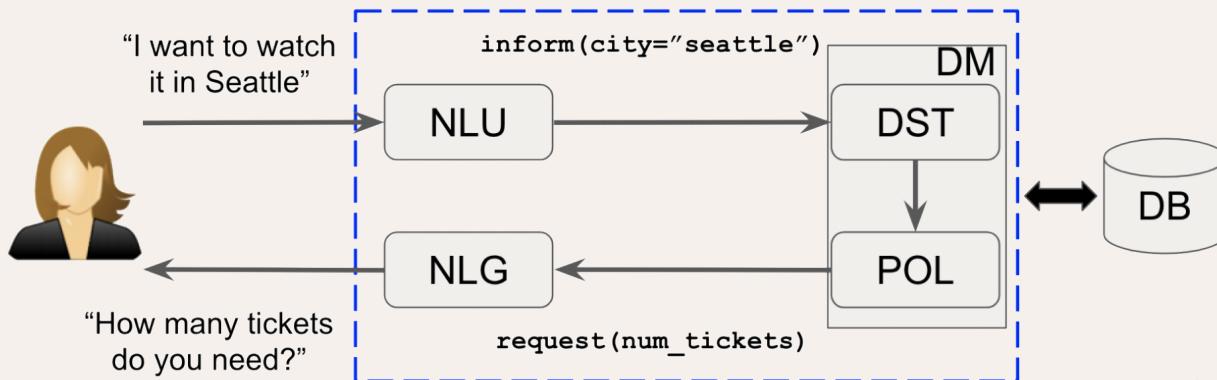
Dialogue State Tracking:

- Содержит список слотов, которые заполняются по ходу диалога
- NLU распарсивает слоты (slot filling) и передает их в DST
- А тот их распихивает по нужным местам
- Dialogue State Tracking Challenges –  
бенчмарк для DST



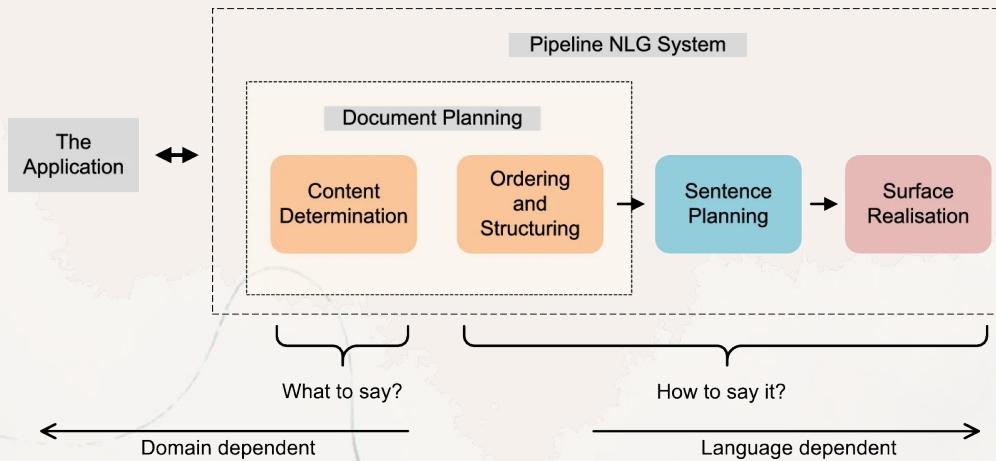
# Task-oriented DS

- Policy learning позволяет боту выбрать, что ему ответить
- Обычно этот модуль обучаю либо с учителем, либо с подкреплением



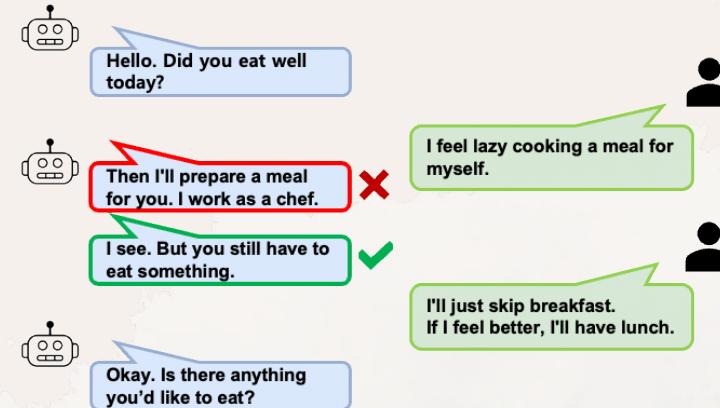
# Task-oriented DS

- NLG должен сгенерировать ответ из информации, которую ему предоставляет PL  
(схемка очень старая)
- Сегодня обычно NLG – это End-to-End система, сразу генерирует что надо



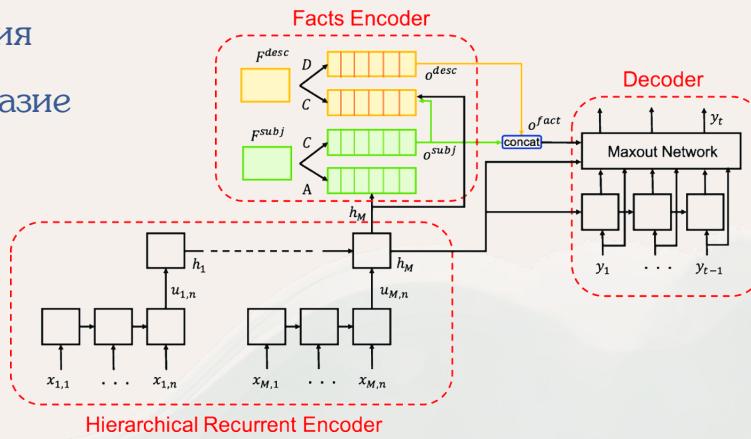
# Open-domain DS

- Подразделяются на три типа:
  - generative systems (генерируют ответы)
  - retrieval-based systems (извлекают готовые ответы из базы данных)
  - ensemble systems (комбинируют генерацию и извлечение)

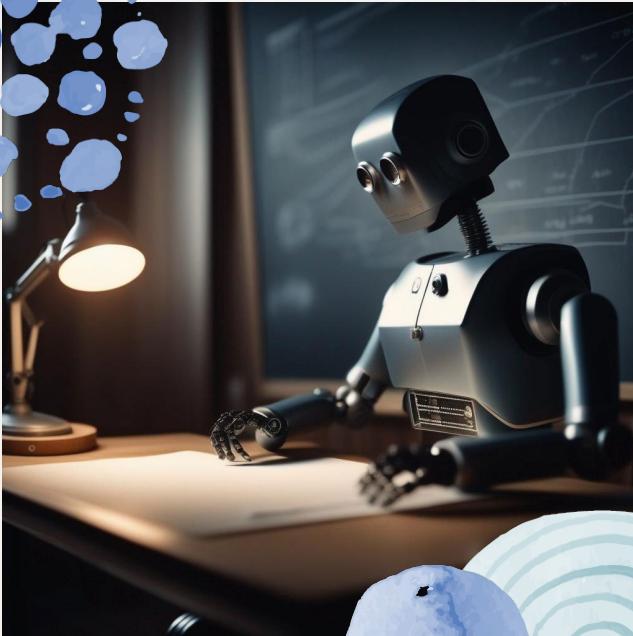


# Open-domain DS

- Должны учитывать контекст (context awareness) – то есть помнить историю диалога
- Для этого используется архитектура Hierarchical Recurrent Encoder-Decoder (HRED)
- В этой архитектуре два энкодера – один кодирует слова, а другой собирает информацию за «ход» диалога и передает ее дальше
- Другая важная проблема – связность ответов (response coherence), существует много разных (сложных) техник для ее улучшения
- Также приходится бороться за разнообразие ответов (response diversity)
- И моделировать характер бота (это уже психолингвистика)



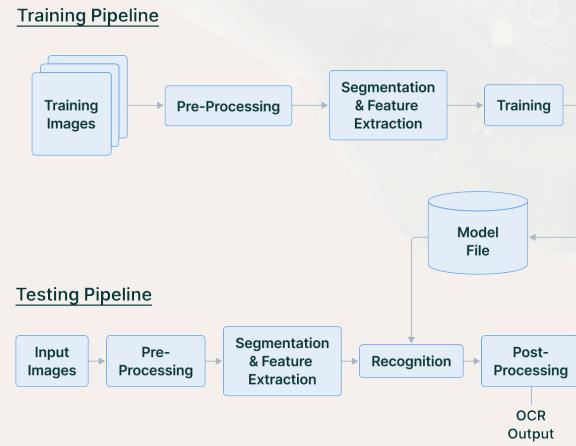
# NLP



# OCR

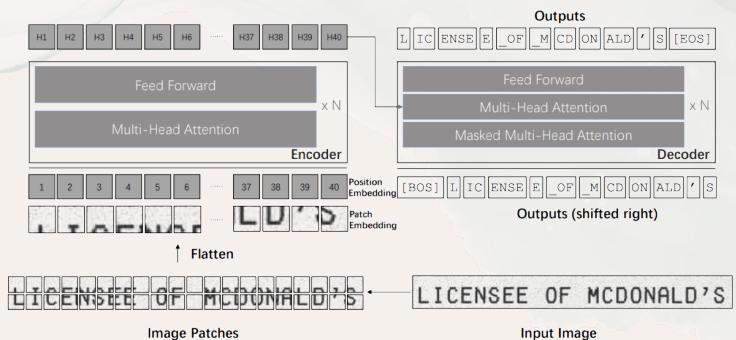
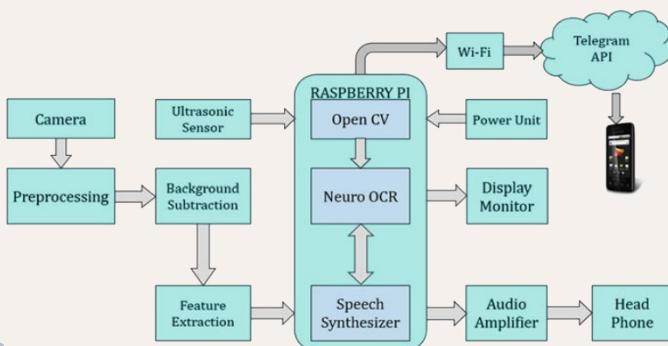
- Оптическое распознание печатного и письменного текста
- Основные шаги модели OCR:
  - Предобработка изображений (убираем шум)
  - Сегментация (выделяем символы)
  - Извлечение признаков (самое важное!)
  - Обучаем алгоритм-классификатор
  - Постобработка результатов (пробуем исправлять ошибки, ориентируясь на контекст)

## General OCR model



# OCR

- Постепенно переходим на архитектуру Transformer:  
TROCR использует ее для распознавания текста
- Качество распознавания рукописного текста достигает 90%



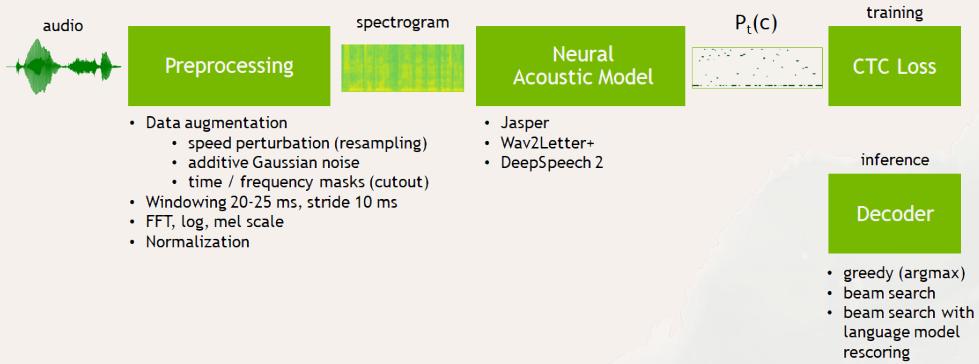
- Разрабатываются приложения, например, для помощи слепым людям, которые распознают тексты и озвучивают их с помощью алгоритма Text-to-Speech

# ASR

- Автоматическое распознание речи

- Подходы:

- Акустический фонетический (1967)
- С распознанием паттерна (1993)
- С использованием ИИ



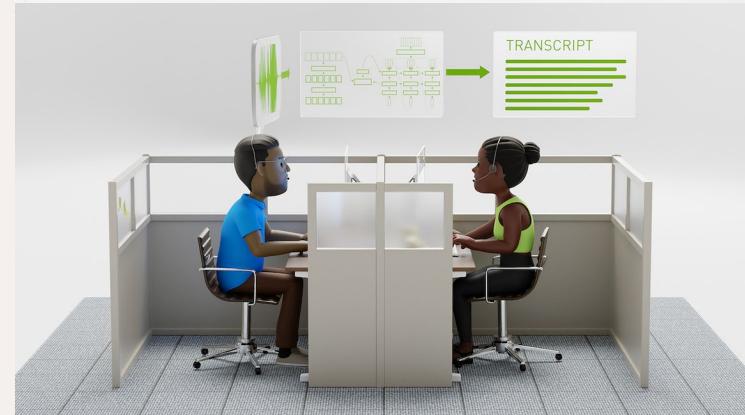
# ASR

- Процесс:
  - Получение звукового сигнала
  - Извлечение признаков
  - Акустическое моделирование (сеть распознает фонемы)
  - Языковое и лексическое моделирование (сеть складывает фонемы в слова)



# ASR

- Люди допускают в среднем около 12% ошибок при распознании устной речи: Deep Speech 2: End-to-End Speech Recognition in English and Mandarin
- Современные модели достигают значений до 2% ошибок
- Последние разработки в этой области – улучшение транскрипции, исправление ошибок распознания (пост-обработка), улучшение качества распознания шумных данных
- Специалисты в этой области говорят, что до совершенства осталось недалеко



# ASR

- Простая в использовании библиотека – SpeechRecognition

```
!pip install pyaudio
```

```
!pip install SpeechRecognition
```

```
import speech_recognition as sr
```

```
AUDIO_FILE = "audio.wav" # путь к файлу-источнику
```

```
r = sr.Recognizer()
```

```
with sr.AudioFile(AUDIO_FILE) as source:
```

```
    audio = r.record(source)
```

```
# читает файл в свой внутренний формат
```

```
print(r.recognize_google(audio, language='en'))
```

```
r = sr.Recognizer()
```

```
with sr.Microphone() as source:
```

```
("Я вас слушаю!") audio = r.listen(source, 5, 15) # ждать 5 сек, слушать 15 сек
```