Hyper speed in large language modeling

Ondřej Dušek, Petr Schwarz, Ondřej Plátek, Santosh Kesiraju

JSALT Summer School

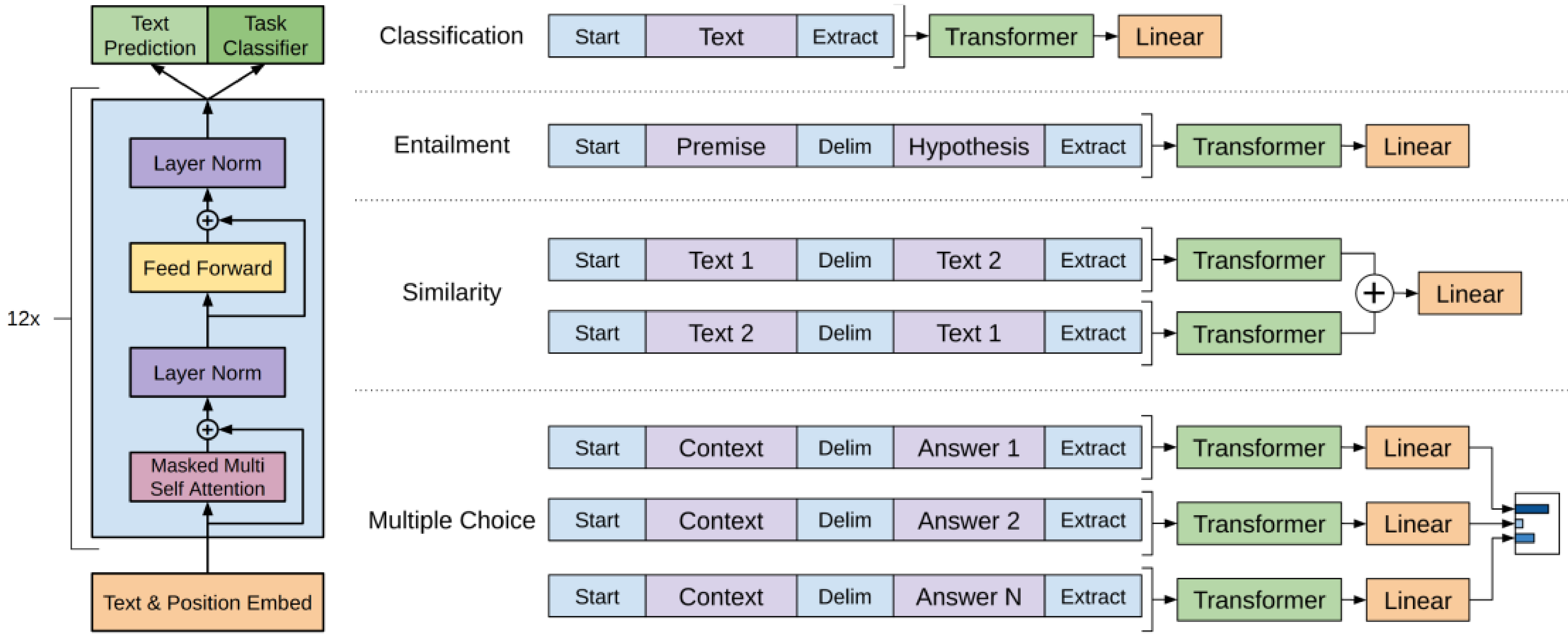
20 June 2022

Ondrej Platek is supported by
Charles University project GAUK 40222
& EU H2020 project no. 952026, HumanE-AI-Net

Topics

- GPT
- Instruct GPT / Chat GPT
- Timeline of LLMs
- Hugging Face Leaderboard
- Low Rank Adaptation (LoRa)
- Quantized LoRa

GPT - Generative Pre-trained Transformer



GPT - Generative Pre-trained Transformer

- Unsupervised pretraining

  Corpus of tokens $\mathcal{U} = \{u_1, \ldots, u_n\}$

  Implementation

  $$h_0 = UW_e + W_p$$
  $$h_l = \texttt{transformer\_block}(h_{l-1}) \forall i \in [1, n]$$
  $$P(u) = \texttt{softmax}(h_n W_e^T)$$

  Context vector $U = (u_{-k}, \ldots, u_{-1})$

  Objective

  $$L_1(\mathcal{U}) = \sum_i \log P(u_i | u_{i-k}, \ldots, u_{i-1}; \Theta)$$

- Supervised fine-tuning

  Linear layer and Softmax added $P(y|x^1, \ldots, x^m) = \texttt{softmax}(h_l^m W_y)$

  Objective

  $$L_2(\mathcal{C}) = \sum_{(x,y)} \log P(y|x^1, \ldots, x^m)$$

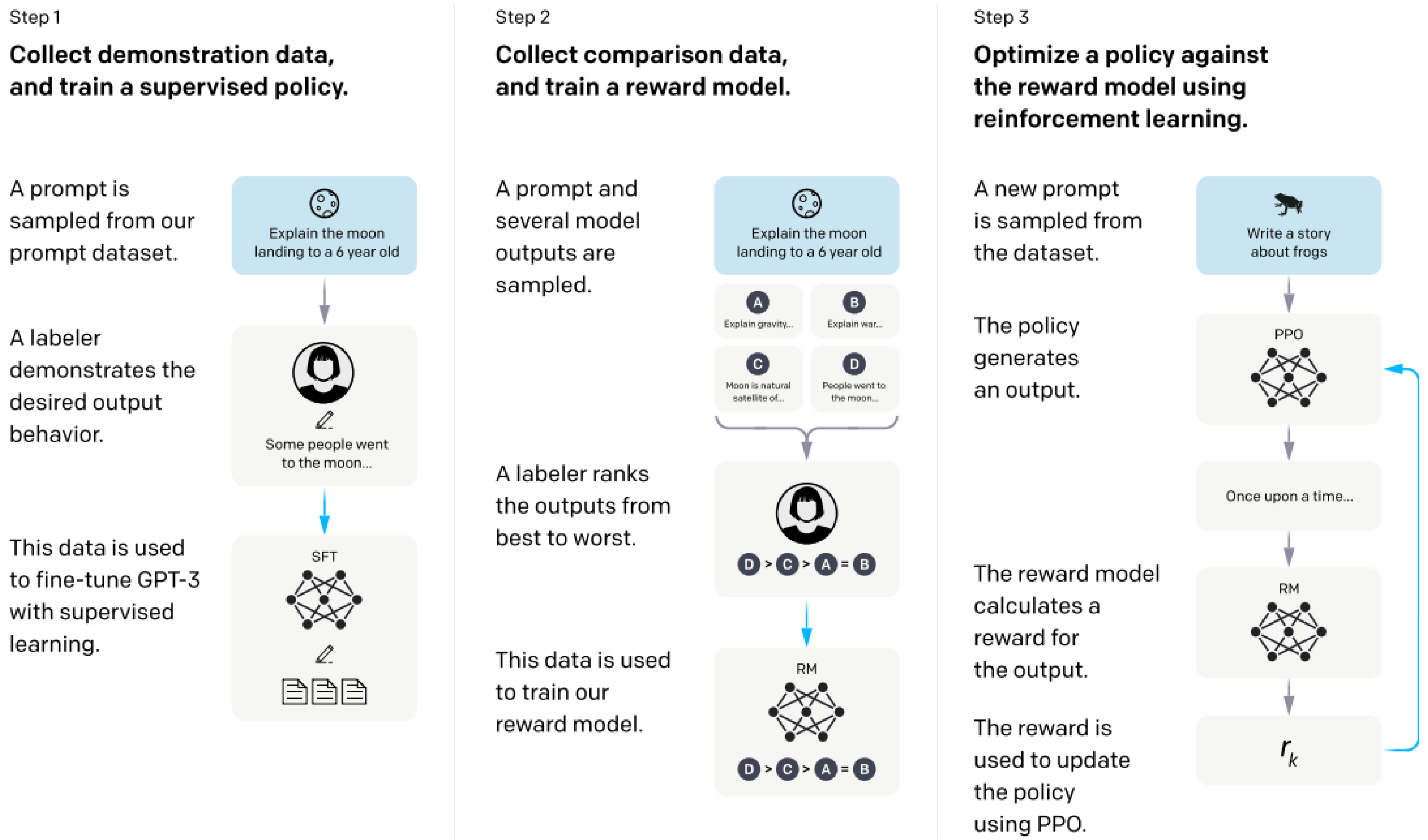
  Adding unsupervised criteria helps

  $$L_3(\mathcal{C}) = L_2(\mathcal{C}) + \lambda * L_1(\mathcal{C})$$

GPT and model size

• GPT 1	120M
• GPT 2	1.5B
• GPT 3	175B

Instruct GPT / Chat GPT



Step 1
Collect demonstration data, and train a supervised policy.
A prompt is sampled from our prompt dataset.
Explain the moon landing to a 6 year old
A labeler demonstrates the desired output behavior.
Some people went to the moon...
This data is used to fine-tune GPT-3 with supervised learning.
SFT
Step 2
Collect comparison data, and train a reward model.
A prompt and several model outputs are sampled.
Explain the moon landing to a 6 year old
A Explain gravity...
B Explain war...
C Moon is natural satellite of...
D People went to the moon...
A labeler ranks the outputs from best to worst.
D > C > A = B
This data is used to train our reward model.
RM
D > C > A = B
Step 3
Optimize a policy against the reward model using reinforcement learning.
A new prompt is sampled from the dataset.
Write a story about frogs
The policy generates an output.
PPO
Once upon a time...
The reward model calculates a reward for the output.
RM
The reward is used to update the policy using PPO.
r_k

Long Ouyang, et. al.: "Training language models to follow instructions with human feedback", https://arxiv.org/abs/2203.02155

Instruct GPT / Chat GPT

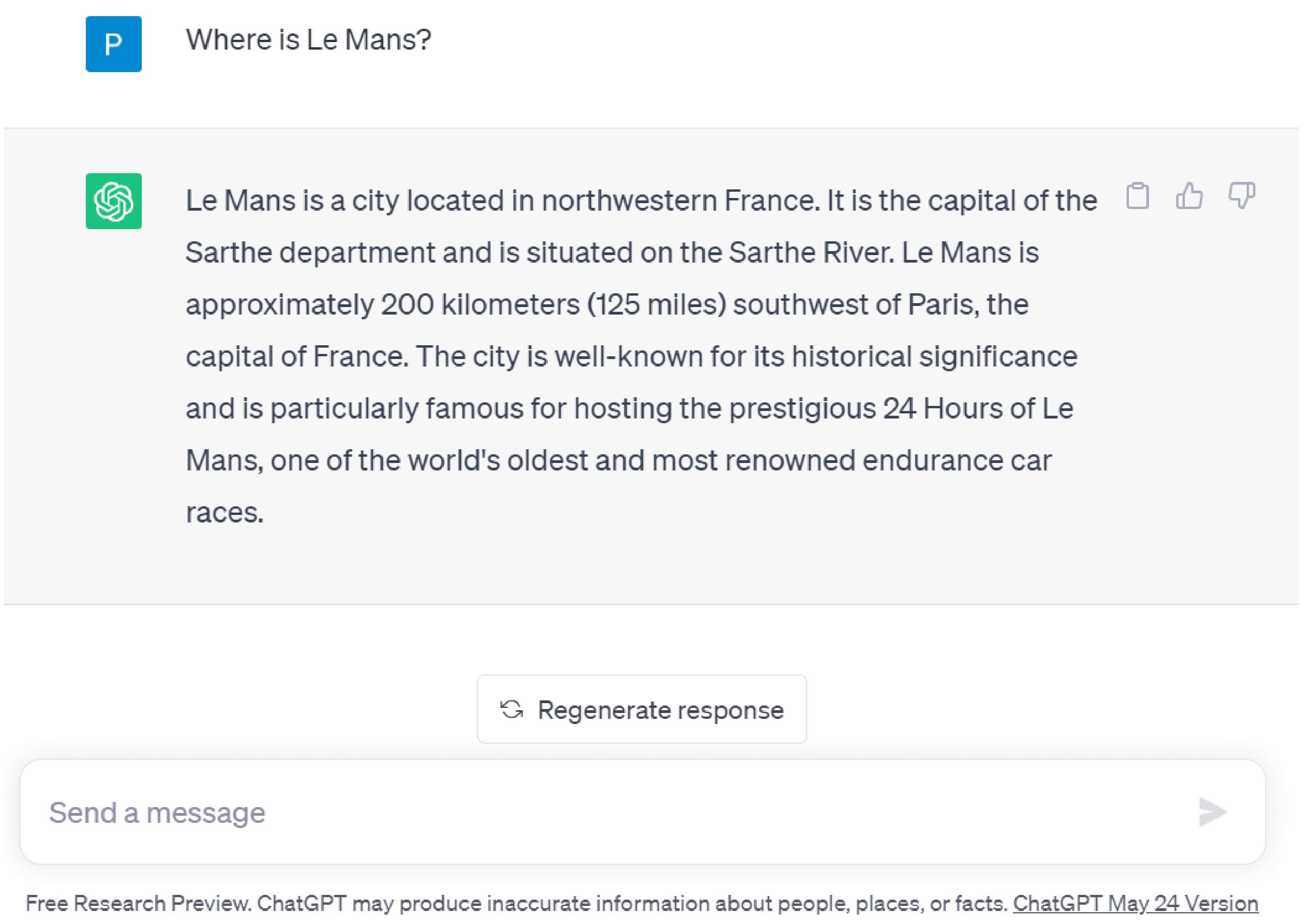
- Labelers were asked to write three kinds of prompts:

  1) Plain: come up with an arbitrary task, while ensuring the tasks had sufficient diversity

  2) Few-shot: come up with an instruction, and multiple query/response pairs for that instruction

  3) User-based: come up with prompts corresponding to use cases requested in OpenAI API

Chat GPT



P

Where is Le Mans?

Le Mans is a city located in northwestern France. It is the capital of the Sarthe department and is situated on the Sarthe River. Le Mans is approximately 200 kilometers (125 miles) southwest of Paris, the capital of France. The city is well-known for its historical significance and is particularly famous for hosting the prestigious 24 Hours of Le Mans, one of the world's oldest and most renowned endurance car races.

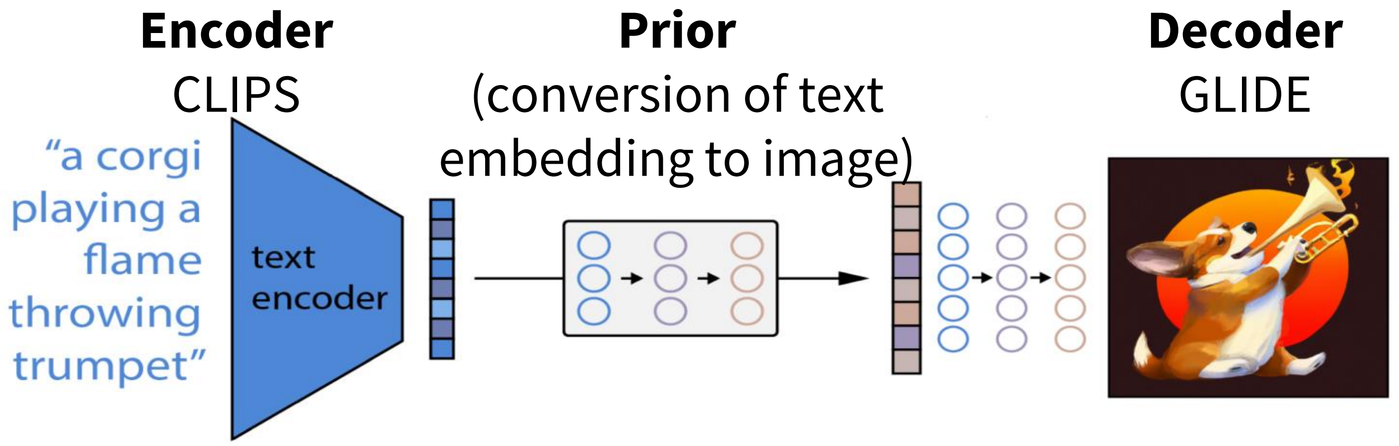
Free Research Preview. ChatGPT may produce inaccurate information about people, places, or facts. ChatGPT May 24 Version

LLM Timeline - Chat GPT

- OpenAI released Chat GPT on November 30th and it raised a wave of interest in general public
- All the school kids are using Chat GPT to help with their school assignments
- Microsoft implemented it into BING for general search
- Similar trend is in other fields (graphic) - changing of artistic style; or in multidomain areas (text - graphic) - generation of graphics from textual prompts
- Training is very costly ~ millions of USD

Open AI Dall-E

a bowl of soup that
is a portal
to another dimension



https://openai.com/dall-e-2

https://www.assemblyai.com/blog/how-dall-e-2-actually-works/

LLM Timeline – LLaMa

- Meta released LLaMa February 24th 2023

  - 7B, 13B, 33B, 65B parameters versions
  - code available
  - weights for noncommercial purposed
  - no instruction finetuning
  - very powerful model for research –> brought strong push to our field

- What happened then is well summarized by people from Google (marked as leaked communication)

  - Dylan Patel and Afzal Ahmad: "Google "We Have No Moat, And Neither Does OpenAI“, https://www.semianalysis.com/p/google-we-have-no-moat-and-neither

LLM Timeline

- March 12th Artem Andreenko managed to run the model on Raspberry Pi
- March 13th Alpaca is released

  - instruction fine-tuning of LLaMa
  - done on single RTX 4090 in hours using the LoRa technique (Low Rank Adaptation)
- March 18th Georgi Gerganov created 4-bit quantization version of LLaMa

  - runs fast on no GPU computers now
- March 19th Vicuna model

  - 13B parameters
  - finetuning of LLaMa on user-shared conversations from SharedGPT web side
  - 300 USD cost

LLM Timeline

- March 25th - GPT4All - a package of models for download to your laptop
- March 28th - Cerebras model

  - "open source GPT-3"
  - optimal compute schedule - number of tokens per parameter to reach optimal utilization of training HW
  - training tricks (reparameterization)
  - the community no longer depends on LLaMa

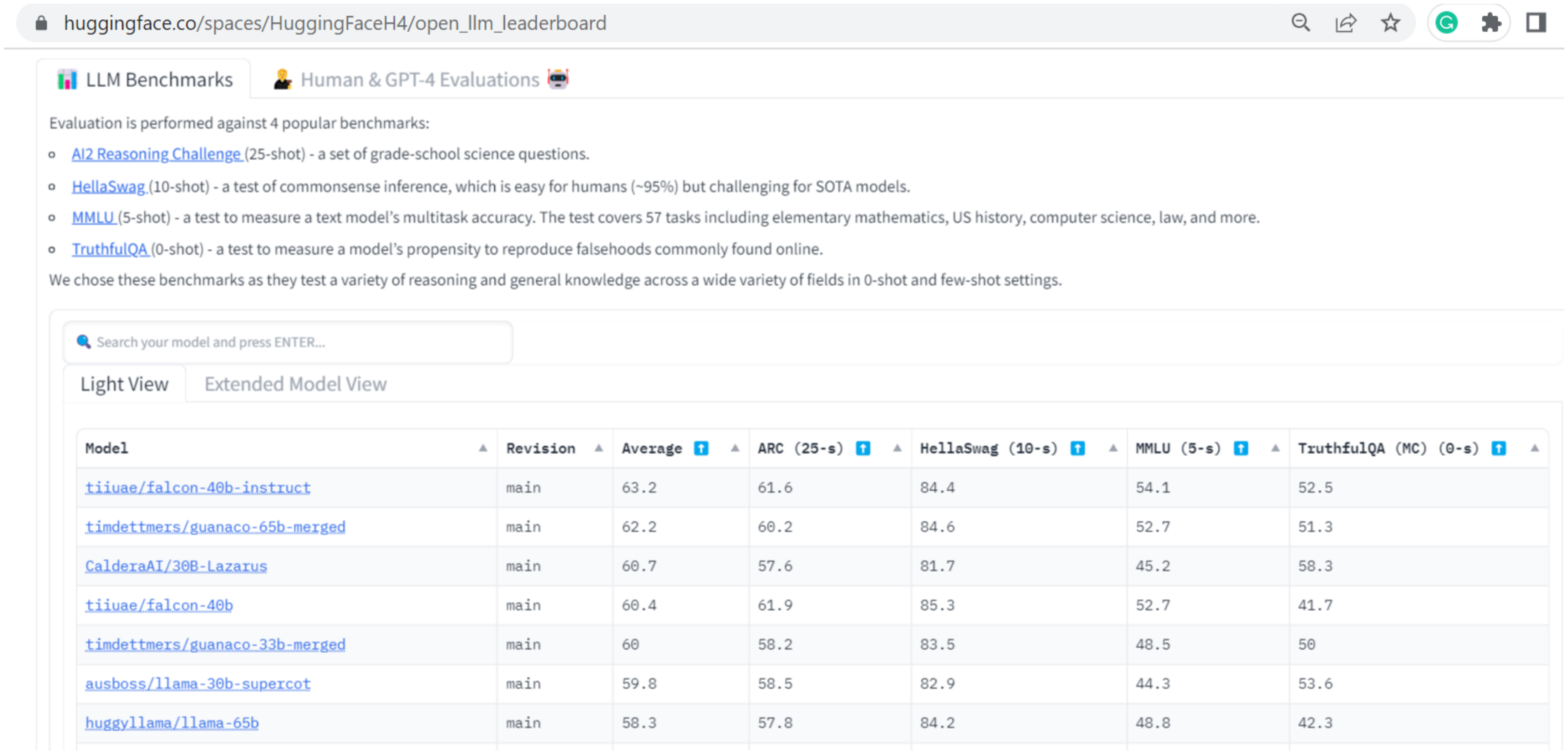
- March 28th - Multimodal training in 1 hour

  - Parameter Efficient Fine Tuning (PEFT), LLaMa-Adapter
  - only 1.2M trainable parameters

LLM Timeline

- April 3rd - Berkley lunches Koala
  - trained entirely on free data
  - cost 100 USD
- April 15th - Open Assistant
  - Reinforcement Learning from Human Feedback (like ChatGPT)

Hugging Face Leaderboard



huggingface.co/spaces/HuggingFaceH4/open_llm_leaderboard

LLM Benchmarks | Human & GPT-4 Evaluations

Evaluation is performed against 4 popular benchmarks:

- AI2 Reasoning Challenge (25-shot) - a set of grade-school science questions.
- HellaSwag (10-shot) - a test of commonsense inference, which is easy for humans (~95%) but challenging for SOTA models.
- MMLU (5-shot) - a test to measure a text model's multitask accuracy. The test covers 57 tasks including elementary mathematics, US history, computer science, law, and more.
- TruthfulQA (0-shot) - a test to measure a model's propensity to reproduce falsehoods commonly found online.

We chose these benchmarks as they test a variety of reasoning and general knowledge across a wide variety of fields in 0-shot and few-shot settings.

Model	Revision	Average	ARC (25-s)	HellaSwag (10-s)	MMLU (5-s)	TruthfulQA (MC) (0-s)
tiiuae/falcon-40b-instruct	main	63.2	61.6	84.4	54.1	52.5
timdettmers/guanaco-65b-merged	main	62.2	60.2	84.6	52.7	51.3
CalderaAI/30B-Lazarus	main	60.7	57.6	81.7	45.2	58.3
tiiuae/falcon-40b	main	60.4	61.9	85.3	52.7	41.7
timdettmers/guanaco-33b-merged	main	60	58.2	83.5	48.5	50
ausboss/llama-30b-supercot	main	59.8	58.5	82.9	44.3	53.6
huggyllama/llama-65b	main	58.3	57.8	84.2	48.8	42.3

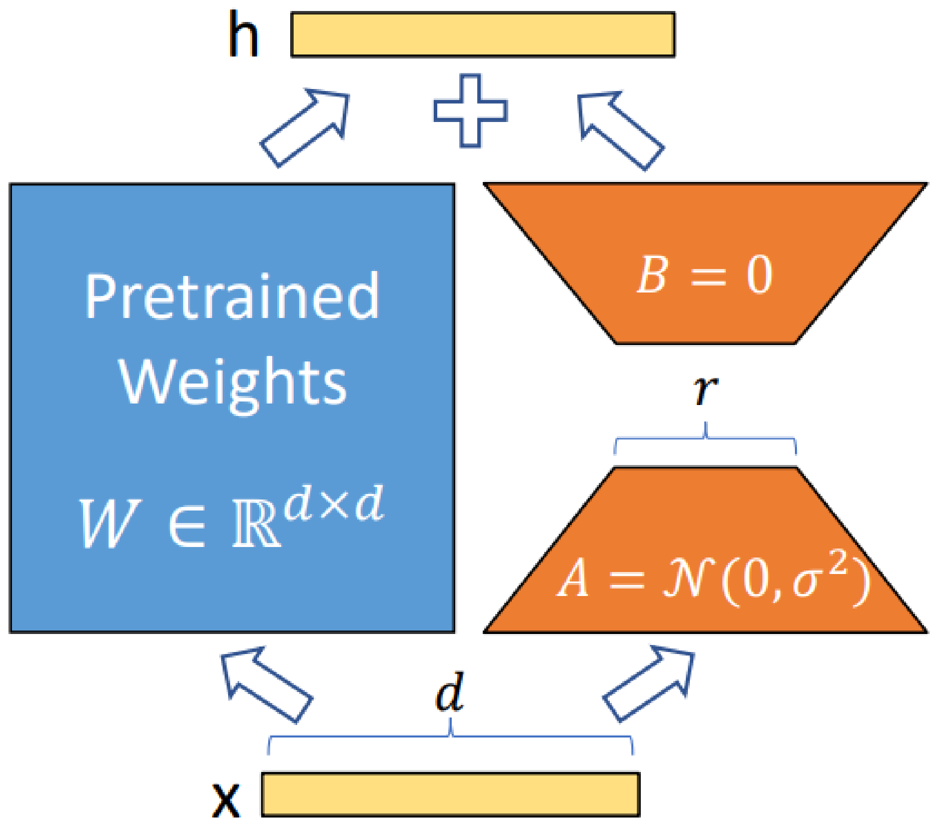
https://huggingface.co/spaces/HuggingFaceH4/open_llm_leaderboard

LoRa – Low Rank Adaptation of Language Models

- Enable us to fine-tune LLM with reasonable costs
- Makes the practical use of LLMs for many downstream tasks possible
- Can reduce the number of trainable parameters by 10 000x
- GPU memory requirements 3x
- No additional latency during inference
- Tested with GPT-3 175B parameters, RoBERTa, DeBERTa, GPT-2 (Microsoft)
- Code available https://github.com/microsoft/LoRA

Edward J. Hu, at al.: "LoRA: Low-Rank Adaptation of Large Language Models", https://arxiv.org/abs/2106.09685

LoRa – Low Rank Adaptation of Language Models


h
Pretrained Weights
$W \in \mathbb{R}^{d\times d}$
$B = 0$
r
$A = \mathcal{N}(0, \sigma^2)$
d
x

- The changes to matrix weights can be stored separately
- It is not necessary to store the wall matrix, it can be factorized to two matrixed that have much lower number of parameters

$$h = W_0 + \Delta W x = W_0 x + BAx$$

- Can be applied to selected matrixes in our transformer that we would like to adapt

Edward J. Hu, at al.: "LoRA: Low-Rank Adaptation of Large Language Models", https://arxiv.org/abs/2106.09685

Quantized LoRa

- Almost not accuracy lost
- LoRA factorized matrixes are stored at 4 bits
- When used it is dequantized (4 bits -> float -> 4 bits for CUDA processing)

  => inference is too slow
- Uses double quantization to reduce the space of quantization constants (without it is +0.5 bit per parameter)
- Page optimizers – when GPU is out-of-memory, the data is moved to CPU RAM

Tim Dettmers, et. al.: "QLoRA: Efficient Finetuning of Quantized LLMs", https://arxiv.org/abs/2305.14314

Flash Attentions instead of Attentions

- Attentions in transformer $Attention(Q, K, V) = softmax(\frac{QK^{\top}}{\sqrt{d_k}})V$
- Input sequence of length n with d dimension embeddings $\boldsymbol{X} \in R^{n\times d}$
- To perform Attention operation in Transformers, we compute

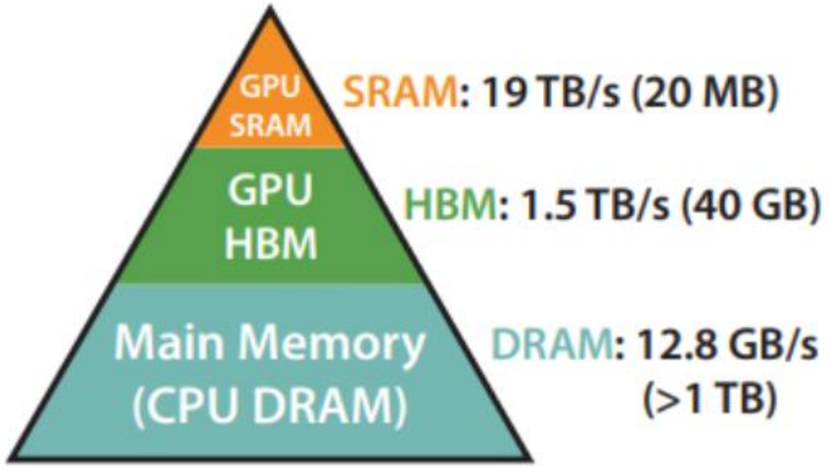
$$\boldsymbol{Q} = \boldsymbol{X}\boldsymbol{W}_Q \qquad \boldsymbol{W}_Q \in R^{d\times d}$$
$$\boldsymbol{K} = \boldsymbol{X}\boldsymbol{W}_K \qquad \boldsymbol{W}_K \in R^{d\times d}$$
$$\boldsymbol{V} = \boldsymbol{X}\boldsymbol{W}_V \qquad \boldsymbol{W}_V \in R^{d\times d}$$

- Pair-wise dot product

$$\boldsymbol{S} = \boldsymbol{Q}\boldsymbol{K}^{\boldsymbol{T}} \qquad \in R^{n\times n} \rightarrow \ O(n^2) \text{ in space}$$

- Row-wise normalization $\boldsymbol{P} = softmax(\boldsymbol{Q}\boldsymbol{K}^{\boldsymbol{T}}) \qquad \in R^{n\times n}$
- Weighted average $\boldsymbol{O} = \boldsymbol{P}\boldsymbol{V} \qquad \in R^{n\times d}$

Flash Attentions instead of Attentions



Memory Hierarchy with Bandwidth & Memory Size

We need:

1. Computing the SoftMax without access to the whole input
2. Not storing the large intermediate attention matrix for the backward pass

- Tiling - split the input into blocks, SoftMax operation incrementally by making several passes over the input blocks.
- Recomputation - The SoftMax normalization factor from the forward pass is stored for backward pass
- Resulting in 7.6x speed up for GPT-2

(Dao et al, 2022) https://arxiv.org/pdf/2205.14135.pdf

Fully Sharded Data Parallel (FSDP)

- Parallel training algorithm
- Distributed Data Parallel (commonly used by PyTorch) copies model (+gradients and optimizer states) to all GPUs
- FSDP decomposes the model instance into smaller units and handles each unit independently
- Can offload data to CPU if needed
- At the cost of increased communication volume.

PyTorch blog: https://pytorch.org/tutorials/intermediate/FSDP_tutorial.html

Model Parallelism / Tensor Parallel

- Consider an MLP with, input $\boldsymbol{X}$, weights $\boldsymbol{W}$, and non-linearity f

$$\boldsymbol{Y} = f(\boldsymbol{X}\,\boldsymbol{W})$$

- Split $\boldsymbol{W}$ along rows $\boldsymbol{W} = \begin{pmatrix} \boldsymbol{W}_1 \\ \boldsymbol{W}_2 \end{pmatrix}$, and $\boldsymbol{X} = (\boldsymbol{X}_1\ \boldsymbol{X}_2)$ along columns

$$\boldsymbol{Y} = f(\boldsymbol{X}_{\boldsymbol{1}}\,\boldsymbol{W}_1 + \boldsymbol{X}_{\boldsymbol{2}}\,\boldsymbol{W}_2)$$

- Split only $\boldsymbol{W}$ along columns $\boldsymbol{W} = (\boldsymbol{W}_1\ \boldsymbol{W}_2)$,

$$\boldsymbol{Y} = [Y_1\ Y_2] = [f(\boldsymbol{X}\,\boldsymbol{W}_1)\ f(\boldsymbol{X}\,\boldsymbol{W}_2)]$$

- Both the choices are conjugate to each other and can be used to speed up some operations.

(Shoeybi et al, 2020) https://arxiv.org/pdf/1909.08053.pdf

Exercise I - finetuning LLMs for task-oriented response generation

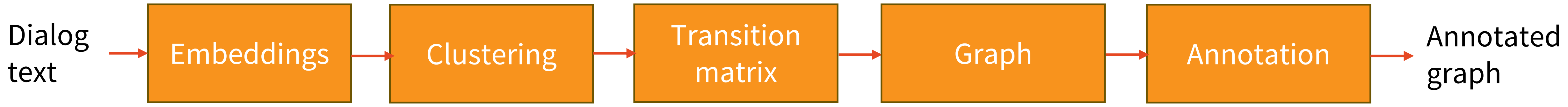
- Prepared by Ondřej Plátek
- GPT2/LLAMA decoder only model
- No DB/API calls
- Next word prediction
- Instruction tuning
- Evaluated on MultiWOZ 2.2
- Quantized LoRa vs. training from scratch

Dialog structure discovery

- Clustering of semantic embeddings
- Using Variational Recurrent Neural Networks
- Using higher level semantic knowledge for clustering (entities, intent tags, API calls …)

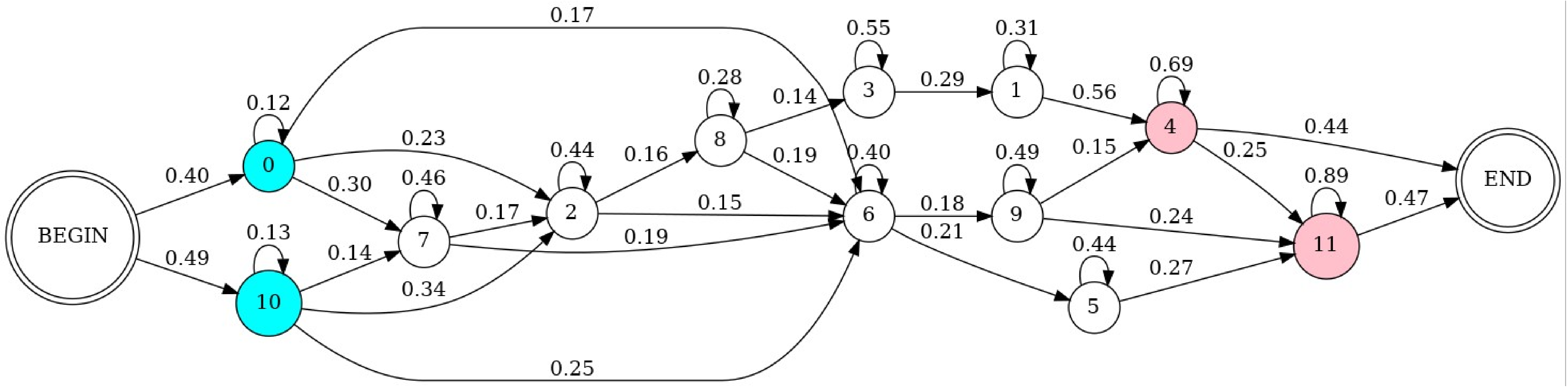
Exercise II - Dialog structure discovery by k-mean clustering

- Prepared by Santosh Kesiraju
- MultiWOZ dataset, restaurant dialogs only
- 14 utterances (7 user-system interactions)
- LaBSE sentence embeddings (variable context – 1 to N utterances)
- k-mean clustering of embeddings
- Transition probability among clusters
- conversion to finite state automata
- Assigning labels to clusters/state based on frequency in the data set



Exercise II - Dialog structure discovery by k-mean clustering

- Example of discovered graph structure on MultuWOZ

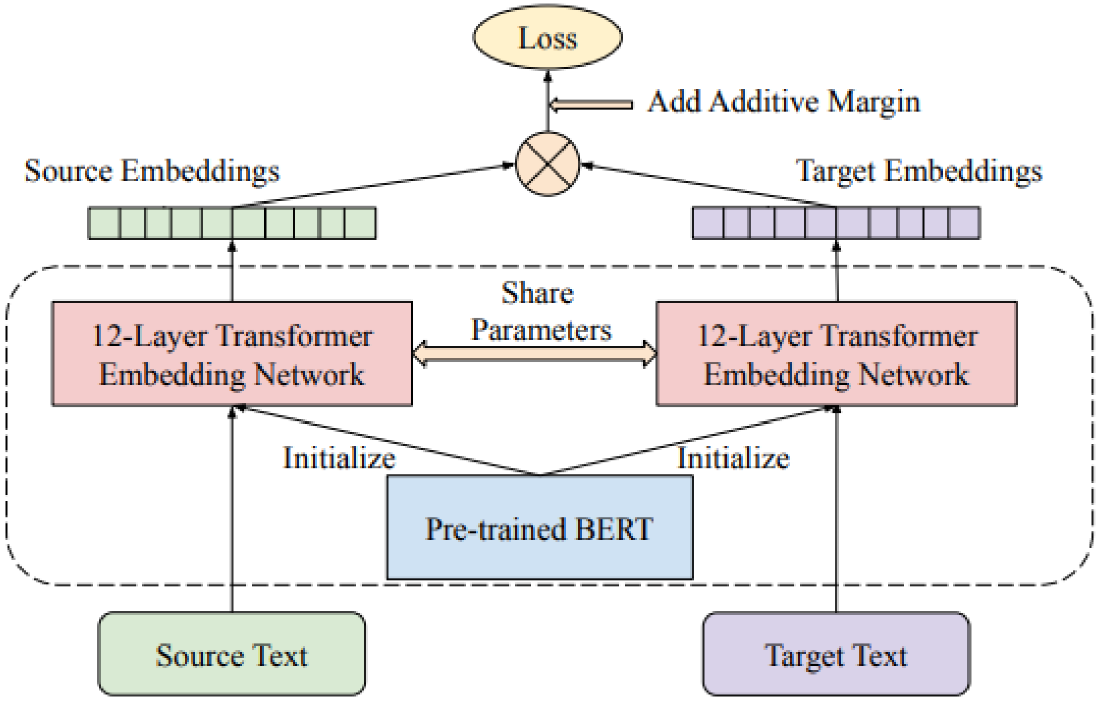


Cluster 3: ill, receipt, completely, slot, bookings, opening, openings, joining, xdyjmto7, 45, hold, wasn, unable, unavailable, monday
Cluster 9: 01223448620, 247877, 01223363471, 01223412299, 01223362525, cb21uf, 01223358966, 01223413000, 01223354755, 01223812660, 01223352500, cb23rh, 367660, 1223, cb41uy,

Language-agnostic BERT Sentence Embedding (LaBSE)

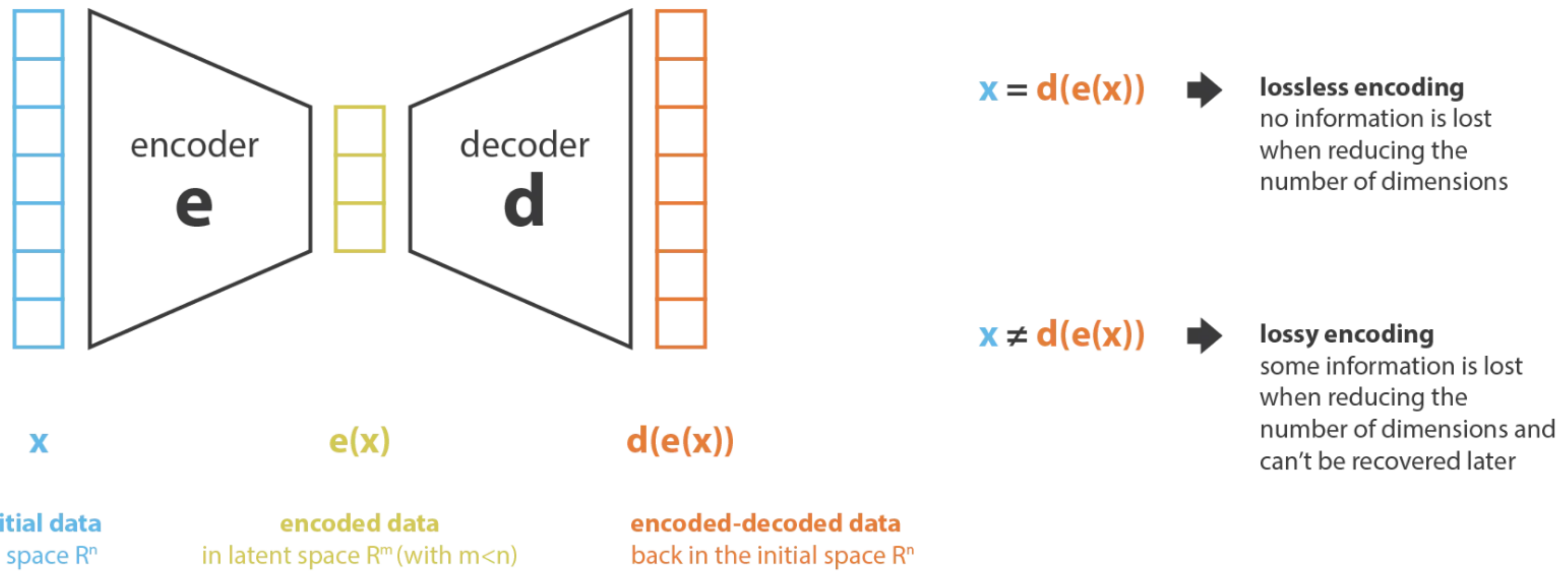
- A publicly released multilingual sentence embedding model spanning 109+ languages
- Pre-training and dual encoder finetuning on translation ranking task
- Loss function: Additive Margin SoftMax

$$\mathcal{L} = -\frac{1}{N}\sum_{i=1}^{N} \frac{e^{\phi(x_i,y_i)-m}}{e^{\phi(x_i,y_i)-m} + \sum_{n=1,n\neq i}^{N} e^{\phi(x_i,y_n)}}$$



Fangxiaoyu Feng, at al.: “Language-agnostic BERT Sentence Embedding”, https://arxiv.org/abs/2007.01852

Autoencoders



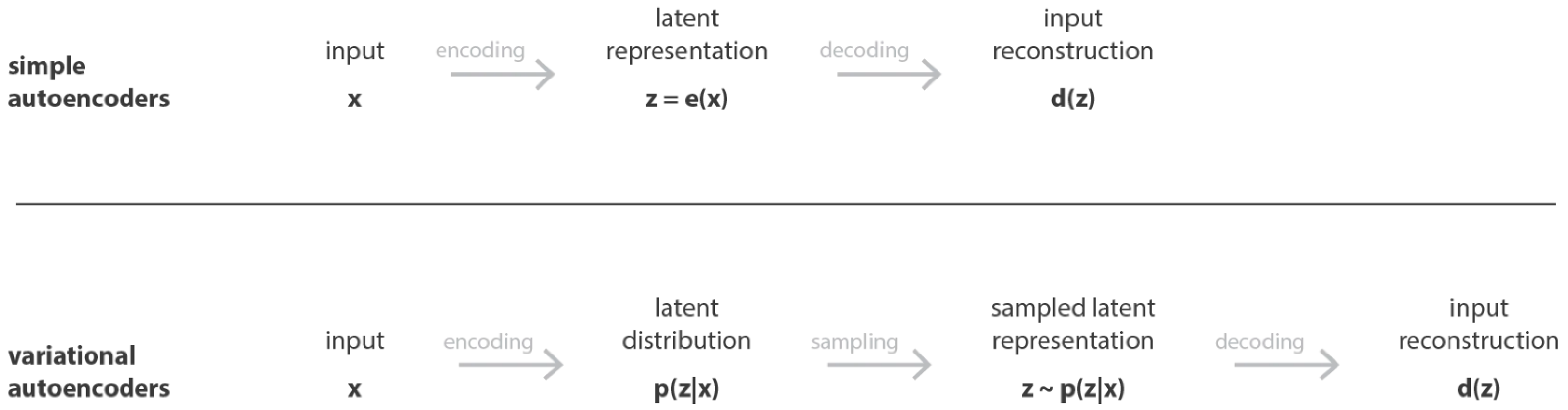
encoder
e
decoder
d
x
e(x)
d(e(x))
initial data
in space R^n
encoded data
in latent space R^m (with m<n)
encoded-decoded data
back in the initial space R^n
x = d(e(x))
lossless encoding
no information is lost when reducing the number of dimensions
x ≠ d(e(x))
lossy encoding
some information is lost when reducing the number of dimensions and can't be recovered later

- Is the latent space well organized?
- What happen if we generate the points in latent space randomly?
- Is not the neural network overfitted? Remembering all the examples?

Joseph Rocca: "Understanding Variational Autoencoders (VAEs)", https://towardsdatascience.com/understanding-variational-autoencoders-vaes-f70510919f73

Variational autoencoders

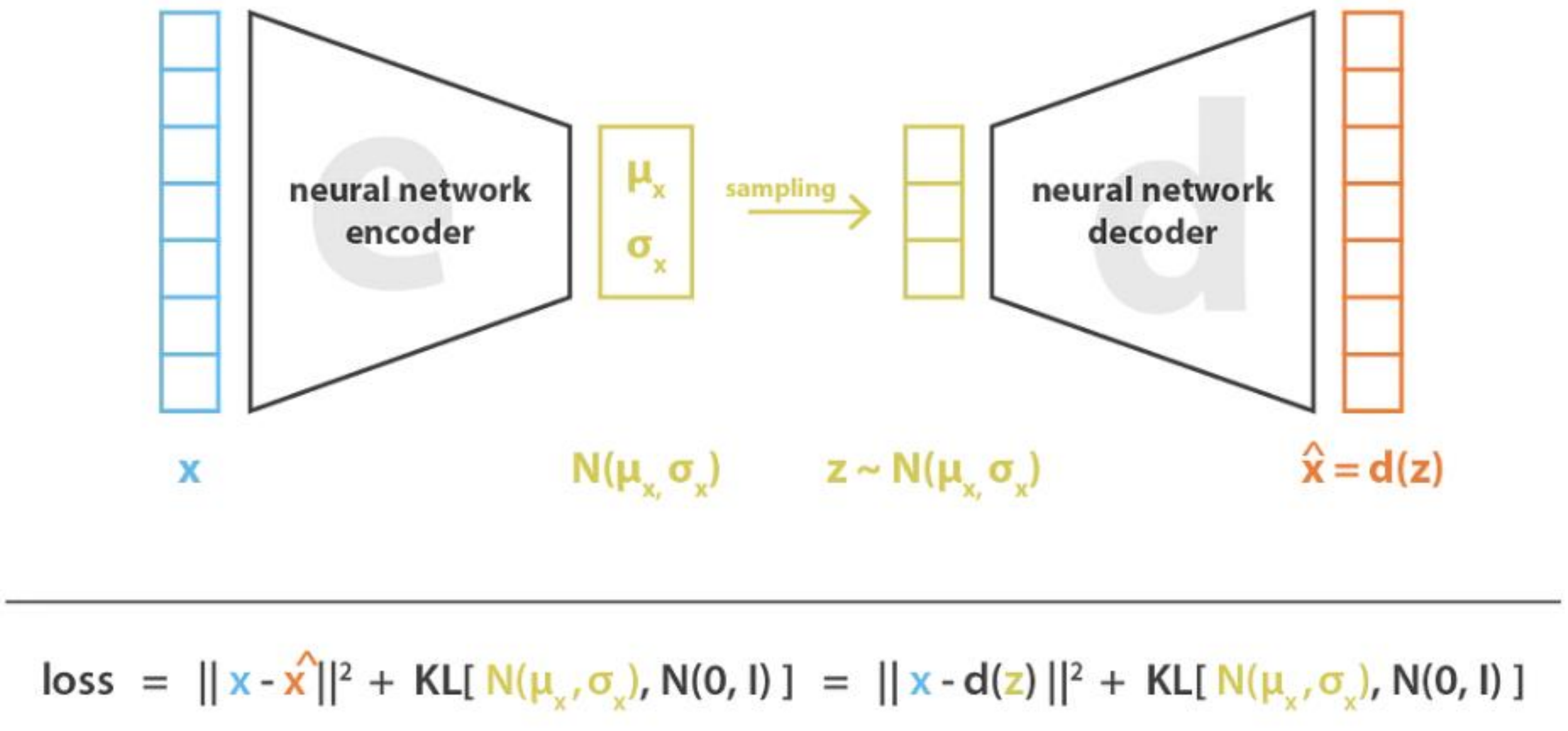
1. The input is encoded as distribution over latent space, for example Gaussian with μ and Σ.
2. A point from the latent space is sampled from the distribution.
3. The point is decoded, and the reconstruction error is calculated.
4. The error is back-propagated through the network



Joseph Rocca: "Understanding Variational Autoencoders (VAEs)", https://towardsdatascience.com/understanding-variational-autoencoders-vaes-f70510919f73

Variational autoencoders

- Can be seen as a regularization

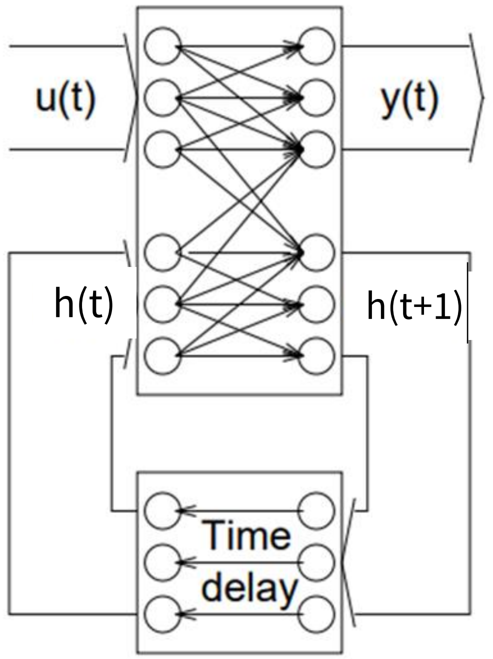


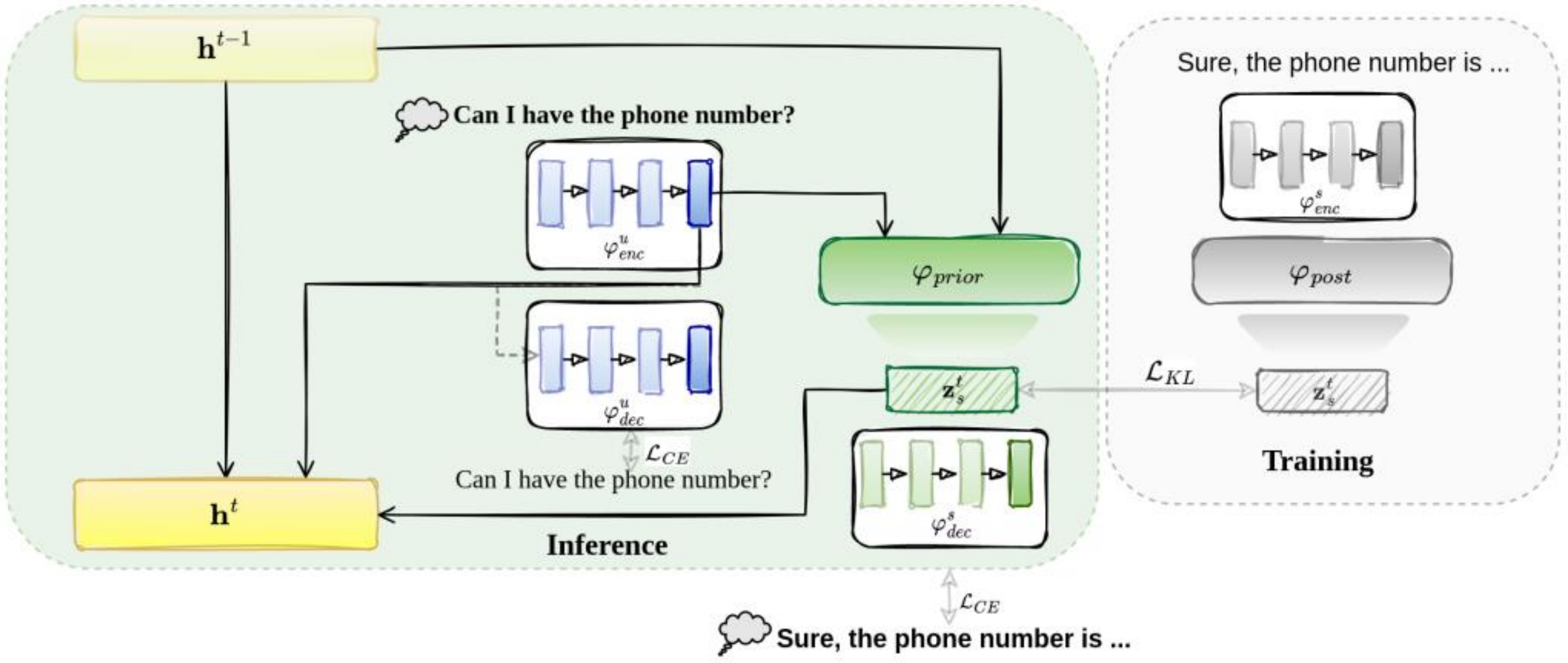
neural network encoder
μ_x
σ_x
sampling
neural network decoder
x
$N(\mu_x, \sigma_x)$
$z \sim N(\mu_x, \sigma_x)$
$\hat{x} = d(z)$
$\text{loss} = \|x - \hat{x}\|^2 + \text{KL}[N(\mu_x, \sigma_x), N(0, I)] = \|x - d(z)\|^2 + \text{KL}[N(\mu_x, \sigma_x), N(0, I)]$

Joseph Rocca: "Understanding Variational Autoencoders (VAEs)", https://towardsdatascience.com/understanding-variational-autoencoders-vaes-f70510919f73

Dialog structure exploration using Variational RNNs

- The posterior distribution is categorical distribution with predefined number of classes
- Gumbel-SoftMax and reparameterization trick is used to get derivatives
- Then it is easy to get the transition matrix again





Vojtěch Hudeček, Ondřej Dušek, “Learning Interpretable Latent Dialogue Actions With Less Supervision”, https://arxiv.org/abs/2209.11128

Evaluation of dialog systems

- Corpus-based evaluation
- Simulator
- Human

Corpus-based evaluation

(Nekvinda & Dušek, 2021)
https://aclanthology.org/2021.gem-1.4

- Task: take real dialogue history from corpus + **generate 1 response**
  - repeat over whole dialogue, collect responses
- Metrics:
  - **Inform rate** – last offered entity matches user constraints
  - **Success rate** – ↑ + system provided all requested information about it
  - **Joint goal accuracy** – % turns where all the slots are captured correctly
  - **BLEU** – n-gram precision (matching sub-phrases of 1-4 words against reference)
- Dialog annotation is needed!
- Problems:
  - really artificial setting, but easiest to use (just need test data)
  - Inf/Succ/JGA: matching the provided entities (more ways to do it)
  - BLEU: tokenization, measuring over delexicalized text

Simulator Evaluation

- **User Simulator** – works as a user, tries to follow goals
- **Dialogue-level** – good over 1 turn ≠ good over whole dialogue
  - especially for end-to-end systems, errors may accumulate over time
  - simulator is the only automatic way to assess this
- Main metric: **Success rate**: was the simulated user's goal reached?
  - i.e. did the system give a correct entity & all information
  - technically same as corpus-based, but now over real dialogues
- Problems:
  - the simulator needs to be built for a given domain
  - it's essentially another dialogue system (🐥 x 🥚)
  - simulator behavior will bias the evaluation

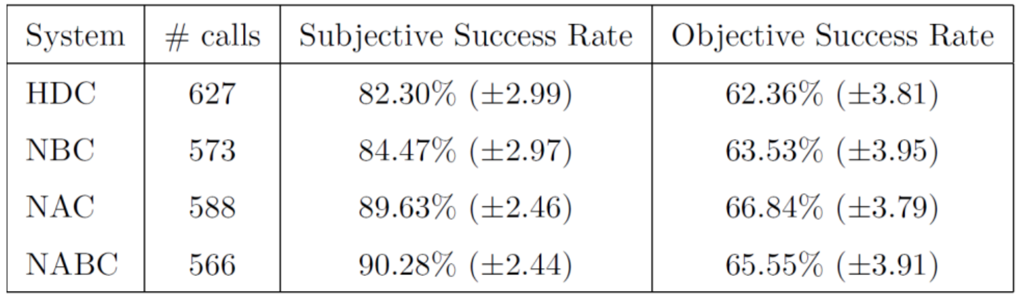
Human Evaluation

Metrics (objective – measuring):

- **Task success** (boolean): did the user get what they wanted?
  - (paid) testers with known goal → check if they found what they were supposed to
    - [warning] sometimes people go off script
  - basic check: did we provide any information at all?
- **Duration**: number of turns (fewer is better)

Metrics (subjective – questionnaries):

- **Success rate:** Did you get all the information you wanted?
  - typically different from objective measures!
- **Future use:** Would you use the system again?
- Component-specific questions



System	# calls	Subjective Success Rate	Objective Success Rate
HDC	627	82.30% (±2.99)	62.36% (±3.81)
NBC	573	84.47% (±2.97)	63.53% (±3.95)
NAC	588	89.63% (±2.46)	66.84% (±3.79)
NABC	566	90.28% (±2.44)	65.55% (±3.91)

(Jurčíček et al., 2012)
https://doi.org/10.1016/j.csl.2011.09.004

Questions?

If you have any question, come or reach us on Discord.