Manifold Learning and Artificial Intelligence

Lecture 11

A Path toward AGI

Extractive Summarization as Feature Selection

Momiao Xiong, University of Texas School of Public Health

- Time: 10:00 pm, US East Time, 03/18/2023
- 10:00 am, Beijing Time. 03/19/2023

Github Address: https://ai2healthcare.github.io/

压缩即泛化，泛化即智能，大模型, 孙思明

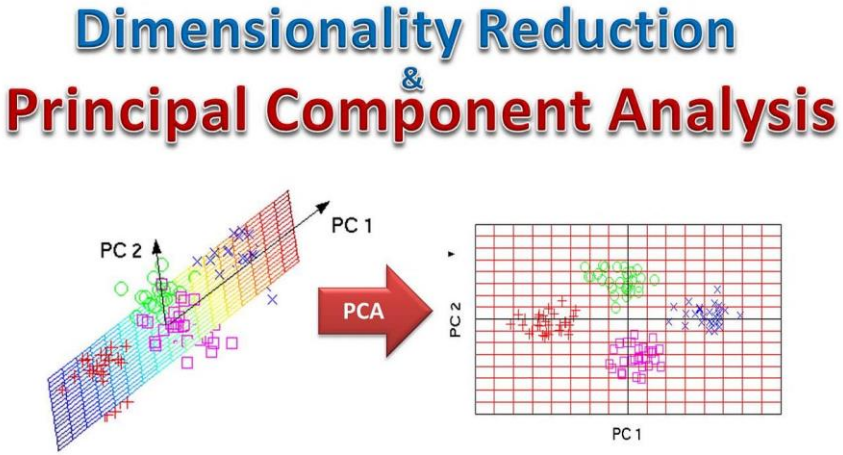
- 1. 通用人工智能（AGI）的追求在于更强的泛化能力。泛化能力越强，智能水平越高。
- 2.压缩就是泛化。对于一个数据集最好的无损压缩，就是对于数据集之外的数据最佳泛化。
- 3.GPT预测下一个token的训练任务，等同于对训练数据进行无损压缩。GPT是目前最好的数据无损压缩算法，因此具备最强的智能。

Summarization as a New Paradigm for Data Reduction

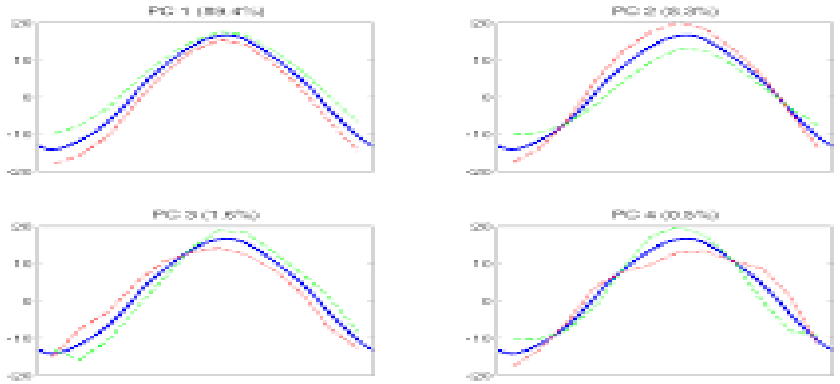
- **Extractive Summarization Approach to Feature Selection**
- **Abstract Summarization Approach to Dimension Reduction**

11. 1. Classical Methods for Data Reduction

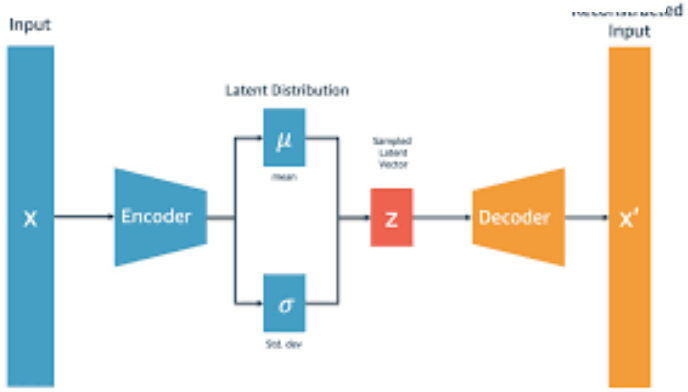
- **PCA**



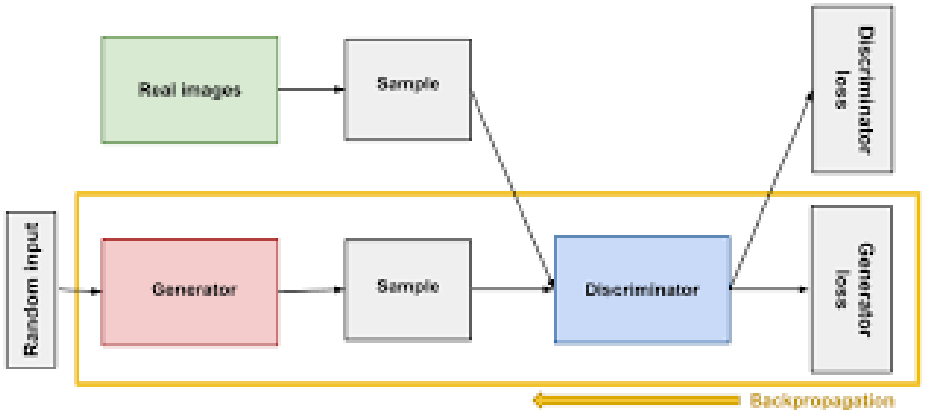
FPCA



- **VAE**



- **GAN**



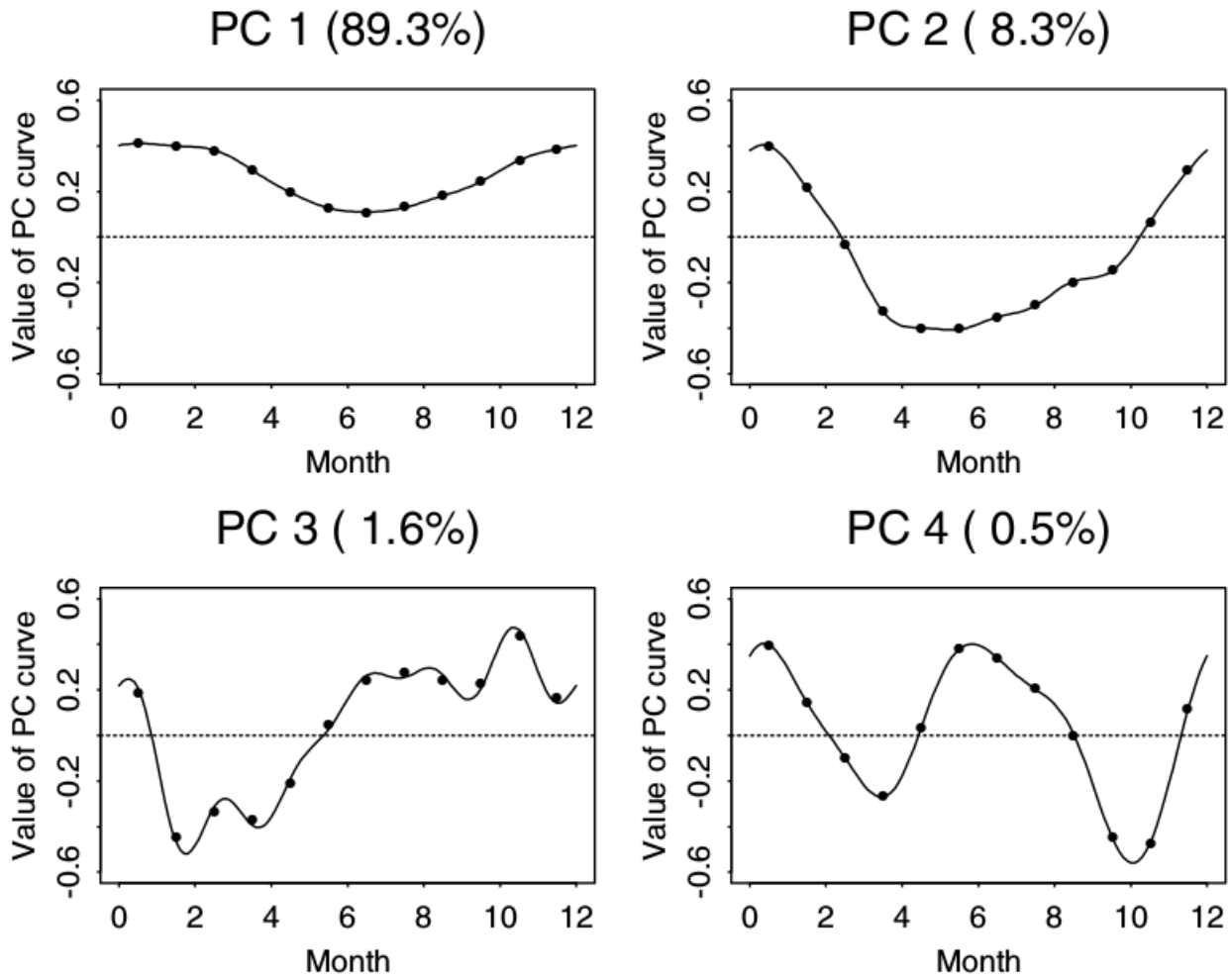


Figure 8.1. The first four principal component curves of the Canadian temperature data estimated by two techniques. The points are the estimates from the discretization approach, and the curves are the estimates from the expansion of the data in terms of a 12-term Fourier series. The percentages indicate the amount of total variation accounted for by each principal component.

11.2. Text Summarization

The goal of summarization is to compress large amounts of information into a shorter, low dimensional format while retaining the most important and relevant contents.

- **Extractive Text Summarization**

  The model “extracts” the most important sentences(or features) from the original text and does not change the sentences (or features).

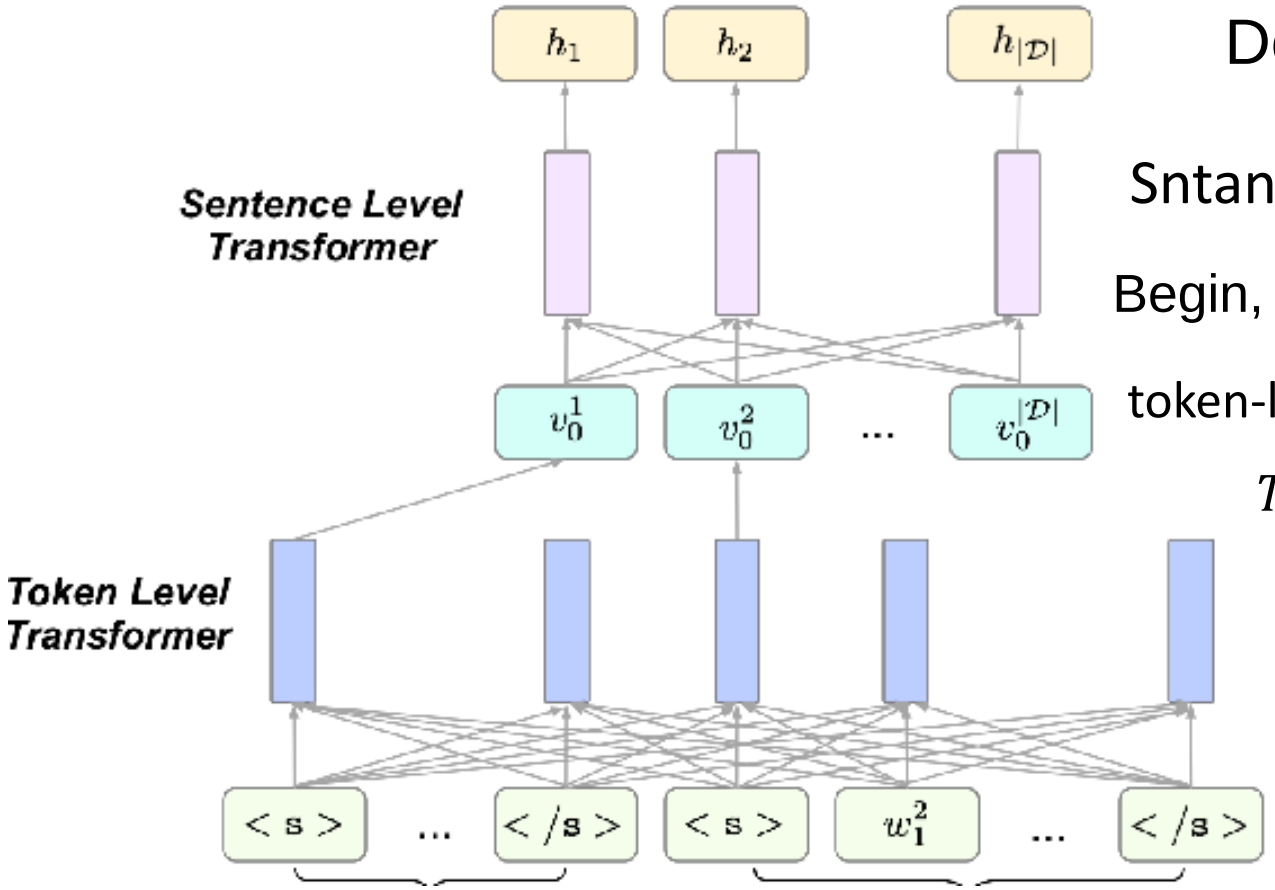
- **Abstract Text Summarization**

  Abstractive Text Summarization converts a collection of text or documents into a short summary that include the important information in the original text ,while may or may not include words and/or sentences from the original text.

Unsupervised Extractive Summarization by Pre-training Hierarchical Transformers

11.3 Extractive Text Summarization

Document Modeling



Document: $D = (S_1, \dots, S_{|D|})$

Sntance: $S_i = \left(w_0^i, w_1^i, \dots, w_{|s_i|}^i\right)$, w_j^i: token.

Begin, end of a sentence: $w_0^i =< s >, w_{|s_i|}^i =</s >$

token-level Transformer: $Trans^T$, sequence level: $Trans^S$

$$Trans^T(D) = (v_0^1, \dots, v_{|s_1|}^1, \dots, v_0^{|D|}, \dots, v_{|s_{|D|}|}^{|D|})$$

$$D = (S_1 || S_2 \quad || S_{|D|})$$

Embedding for sentences in D: $V = (v_0^1, \dots, v_0^{|D|})$

$$H, A = Trans^S(V), H = (h_1, \dots, h_{|D|})$$

h_i: final embedding of S_i, A: Self-attention matrix

A_{ij} : attention score from sentence S_i *to sentence* S_j.

To obtain A, we first average the attention scores across different heads and then across different layers.

11.4. Pre-training

- **Masked Sentences Prediction**

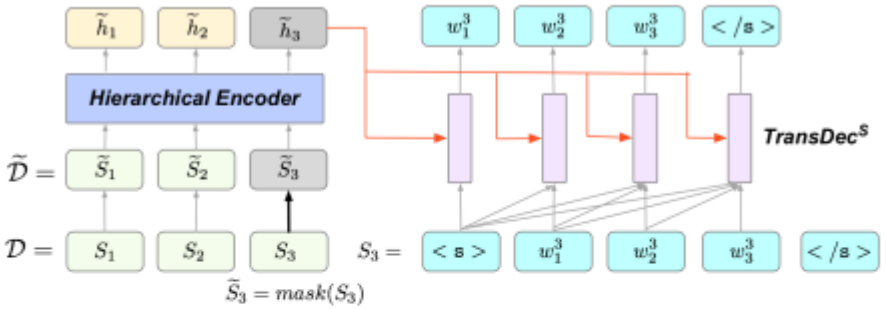


Figure 2: An example of masked sentences prediction. The third sentence in the document is masked and the hierarchical encoder encodes the masked document. We then use $TransDec^S$ to predict the original sentence one token at a time.

$$D = (S_1, \ldots, S_{|D|}), \widetilde{D} = (\widetilde{S_1}, \ldots, \widetilde{S_{|D|}})$$

$$\widetilde{S_i} = \begin{cases} S_i & 85\% \; of \; cases \\ Mask\,(S_i) & 15\% \; of \; cases \end{cases}$$

$$\widetilde{H} = Trans^S(\widetilde{D}), = (\tilde{h}_1, \ldots, \tilde{h}_{|D|})$$

$\tilde{h}_i$: the contextual representation of $\widetilde{S_i}$

Use $\tilde{h}_i$ to predict S_i, one token at a time. Assuming $w^i_{0:j-1}$ has been generated.

$$\tilde{h}^i_j = TransDec^M(w^i_{0:j-1}, \tilde{h}_i) \qquad P\left(w^i_j \middle| w^i_{0:j-1}, \widetilde{D}\right) = softmax\,(W_{vocab} h^i_j)$$

Probability of the original sentences given $\widetilde{D}$ is $\boldsymbol{P(D|\widetilde{D}) = \prod_{S_i \in D} \prod_{j=1}^{|S_i)} P\left(w^i_j \middle| w^i_{0:j-1}, \widetilde{D}\right)}$

- **Sentence Shuffling**

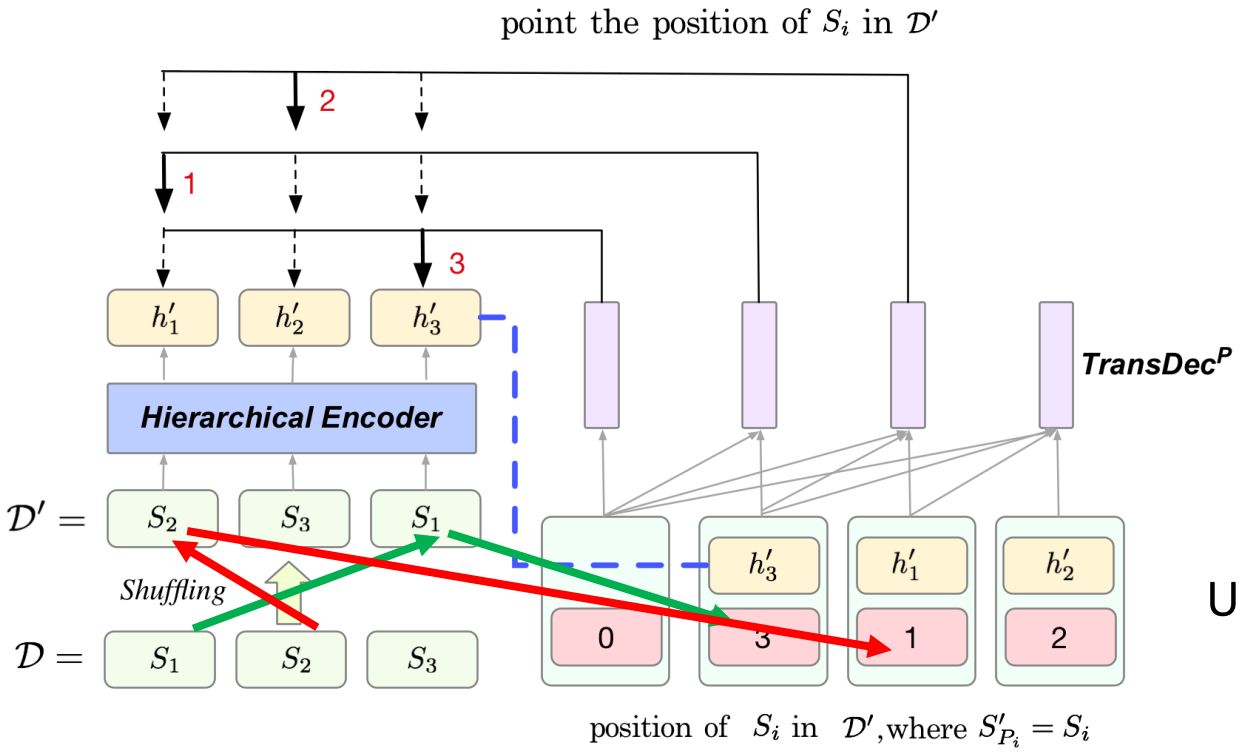


Figure 3: An example of Sentence Shuffling. The sentences in the document are shuffled and then pass through the hierarchical encoder, then a Pointer Network with TransDecP as its decoder is adopted to predict the positions of original sentences in the shuffled document.

Recall $D = (S_1, \ldots, S_{|D|})$.

We permutate the sentences in D and obtain

$D = (S'_1, S'_2, \ldots, S'_{|D|})$, where $S'_{P_i} = S_i$,

P_i is the position in D' .

$H' = (h'_1, \ldots, h'_{|D|})$

Given $P_0, P_1, \ldots, P_{t-1}$, to predict P_t,

Using Point Network Transformer Decoder $Trans\ Dec^P$:

E_{P_i} : Absolute positional embedding of P_i

In original document.

$\boldsymbol{P_i}$: Positional embedding of P_i $during\ decoding$

$h'_1(P_1 = 1), E_{p_1}: embeddin\ of\ P_1 = 1\ in\ original\ Doc$

$M_{t-1} = (h'_{p_1} + \boldsymbol{p_1} + \boldsymbol{E_{P_1}}, \ldots, \boldsymbol{h'_{p_{t-1}}} + \boldsymbol{p_{t-1}} + \boldsymbol{E_{p_{t-1}}})$

Output: $h_t^0 = Trans\ Dec^P(M_{t-1})$

Then the probability of selecting S'_{P_t}:

$p(P_t|P_{1:t-1}, D') = \frac{\exp(g(h_t^0, h'_{p_t}))}{\sum_{i=1}^{|D|} \exp(g(h_t^0, h'_i))}$, where g is the feedforward neural network:

$$g(h_t^0, h'_i) = V_a^T \tanh(U_a h_t^0 + W_a h'_i), \qquad V_a \in R^{d\times 1}, U_a \in R^{d\times d}, W_a \in R^{d\times d}.$$

Finally the probability of positions of original sentences in the shuffled document is:

$$P(\boldsymbol{P}|\boldsymbol{D'}) = \prod_{\boldsymbol{t=1}}^{\boldsymbol{|D|}} \boldsymbol{p(P_t|P_{0:t-1,D'})}$$

During training, for each batch of documents we apply both the masked sentence prediction and sentence shuffling tasks. One document D generates

- **Loss Function:**

$L(\theta) = -[\sum_{D\in\mathcal{X}} \log p(D|\tilde{D}) + \log p(P|D')]$, where $\mathcal{X}$ is the training document set.

11.5. Unsupervised Summarization

In this section, we propose our unsupervised extractive summarization method. Extractive summarization aims to select the most important sentences in document. Once we have obtained a hierarchical encoder using the pre-training methods in Section11.4, we are ready to rank sentences and no additional fine-tuning is needed in this step.

Finding the most important sentence is equivalent to finding the sentence with highest probability . To make the probabilities of different sentences comparable, we normalize them by their length. **Define ranking criterion:**

$$\hat{r}_i = \frac{1}{|S_i|} \sum_{j=1}^{|S_i|} p(w_j^i | w_{0:j-1}^i, D_{\setminus S_i})$$

Normalize it across sentences, finally we obtain **the sentence ranking criterior**:

$$\tilde{r}_i = \frac{\hat{r}_i}{\sum_{j=1}^{|D|} \hat{r}_j}$$

11.6. Modified Sentence Ranking Criterion

- **Second ranking criteria**

  Model the contributions of other sentences to the current sentence explicitly.

  the self-attention matrix of the sentence level Transformer encoder: A

  A_{ji}: Attention score from S_j to S_i

  the second ranking score for S_i

  $$r'_i = \sum_{j=1, j\neq i}^{|D|} A_{ji} \times \tilde{r}_j$$

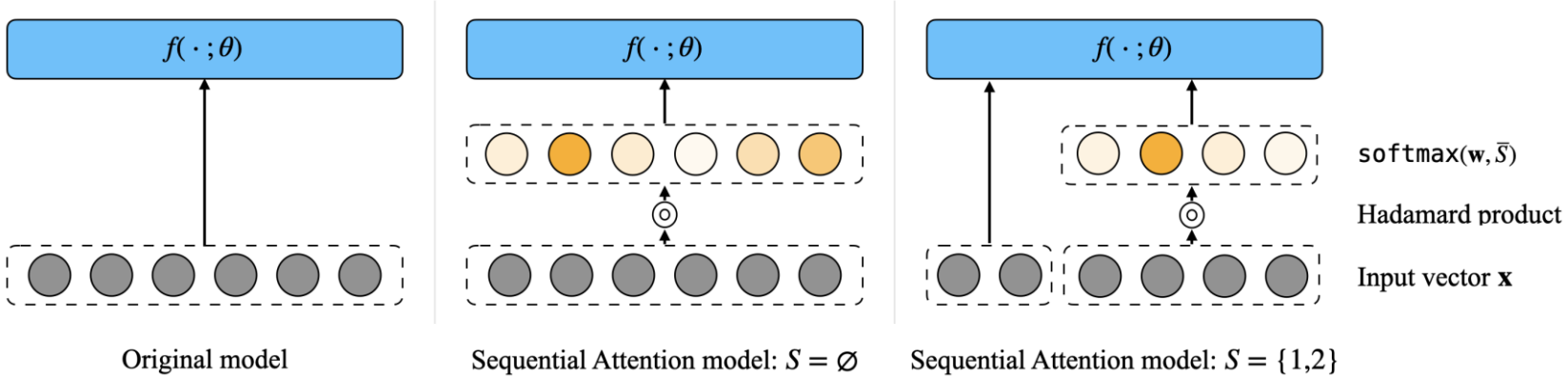
- **Final ranking Score**

  $$r_i = \alpha \tilde{r}_i + \beta r'_i$$

  α, β are coefficients tuned on development set.

  r_i can be computed iteratively by assigning r_i to $\tilde{r}_i$

11.7. Feature Selection



SEQUENTIAL ATTENTION FOR FEATURE SELECTION, Yasuda et al. 2023

The code is available at: github.com/google-research/google-research/tree/master/sequential attention

Algorithm

Step 1: Input sentence embedding matrix $X \in R^{e \times d}$, label $y \in (0,1) or\ y \in R$,neural network for classification $f(X, \theta)$, loss function l, size k, d:the num of feat

Step 2: Initialize $S \leftarrow \emptyset$

For $t = 1$ to k do

$$(\theta^*, w^*) = \underset{\theta, w}{\operatorname{argmin}}\, l(f(X \odot W, \theta), y), W = \mathbf{1}_e softmax(w, \bar{S})^T$$

$$Softmax_i(\mathrm{w}, \bar{S}) = \begin{cases} 1 & i \in S \\ \dfrac{\exp(w_i)}{\sum_{j \in \bar{S}} \exp(w_j)} & i \in \bar{S} = [d] \backslash \mathrm{S} \end{cases} \qquad w = \begin{bmatrix} w_1 \\ \vdots \\ w_d \end{bmatrix},$$

Set $i^* \leftarrow \underset{i \notin S}{arg\max}\, w_i^*$

Update $S \leftarrow S \cup \{i^*\}$

$$Softmax(w, \bar{S}) = \begin{bmatrix} softmax_1(w, \bar{S}) \\ \vdots \\ softmax_1(w, \bar{S}) \end{bmatrix}$$

Return S

Datasets and Architecture

CNN/DailyMail: 287,226 articles for training, 13,368 for validation and 11,490 for test

NYT: 36,745 for training, 5,531 for validation and 4,375 for test.

tokenized with the UTF-8 based BPE tokenizer used in RoBERTa and GPT-2 and the resulting vocabulary contains 50,265 subwords.

Token Level: L = 12;H = 768;A =12.

Sentence Level: L = 6;H = 768;A = 12

4 Nvidia Tesla V100 GPU

ROUGE stands for Recall-Oriented Understudy for Gisting Evaluation. It is essentially a set of metrics for evaluating automatic summarization of texts as well as machine translations. It works by comparing an automatically produced summary or translation against a set of reference summaries (typically human-produced).

Smply put, recall (in the context of **ROUGE**) refers to how much of the reference summary the system summary is recovering or capturing.

Method	CNN/DM			NYT		
	R-1	R-2	R-L	R-1	R-2	R-L
REFRESH (Narayan et al., 2018)	41.30	18.40	37.50	41.30	22.00	37.80
PTR-GEN (See et al., 2017)	39.50	17.30	36.40	**42.70**	**22.10**	**38.00**
BertSumExt (Liu and Lapata, 2019)	**43.25**	**20.24**	**39.63**	–	–	–
BertSumAbs (Liu and Lapata, 2019)	41.72	19.39	38.76	–	–	–
LEAD-3	40.50	17.70	36.70	35.50	17.20	32.00
TEXTRANK (tf-idf)	33.20	11.80	29.60	33.20	13.10	29.00
TEXTRANK (skip-thought)	31.40	10.20	28.20	30.10	9.60	26.10
TEXTRANK (BERT)	30.80	9.60	27.40	29.70	9.00	25.30
PACSUM (Zheng and Lapata, 2019)	40.70	17.80	36.90	41.40	21.70	37.50
PACSUM (BERT) *	40.69	17.82	36.91	40.67	21.09	36.76
PACSUM (RoBERTa) *	40.74	17.82	36.96	40.84	21.28	37.03
Adv-RF (Wang and Lee, 2018)	35.51	9.38	20.98	–	–	–
TED (Yang et al., 2020)	38.73	16.84	35.40	37.78	17.63	34.33
STAS	**40.90**	**18.02**	**37.21**	**41.46**	**21.80**	**37.57**
STAS + PACSUM	**41.26**	**18.18**	**37.48**	**42.42**	**22.66**	**38.50**

Settings	valid set			test set		
	R-1	R-2	R-L	R-1	R-2	R-L
MSP	41.61	18.30	37.92	40.76	17.78	37.03
MSP+SS (STAS)	41.67	18.47	38.00	40.90	18.02	37.21
$\widetilde{r} = 1/\lvert\mathcal{D}\rvert$	41.58	18.43	37.89	40.74	17.88	37.04
$r' = 0$	33.92	12.93	30.99	33.30	12.61	30.33

Methods	valid set			test set		
	R-1	R-2	R-L	R-1	R-2	R-L
CNN/DM						
PACSUM	-	-	-	40.70	17.80	36.90
STAS	41.67	18.47	38.00	40.90	18.02	37.21
STAS + PACUSM	42.20	18.84	38.44	41.26	18.18	37.48
NYT						
PACSUM	-	-	-	41.40	21.70	37.50
STAS	40.36	20.20	36.00	41.46	21.80	37.57
STAS + PACUSM	41.46	21.22	37.05	42.42	22.66	38.50

Table 5: Results of the combination using ROUGE F1 for CNN/DM and NYT

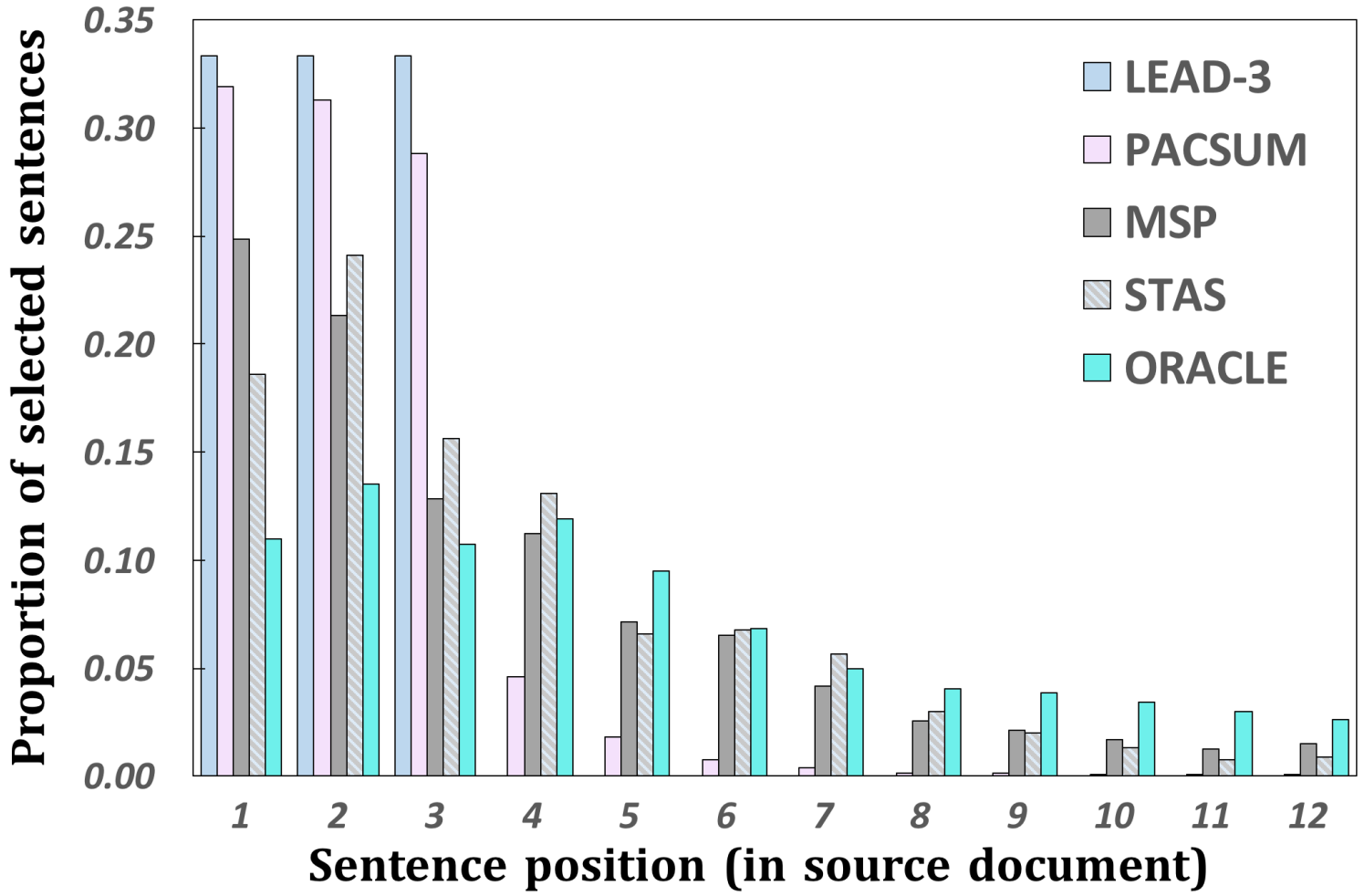
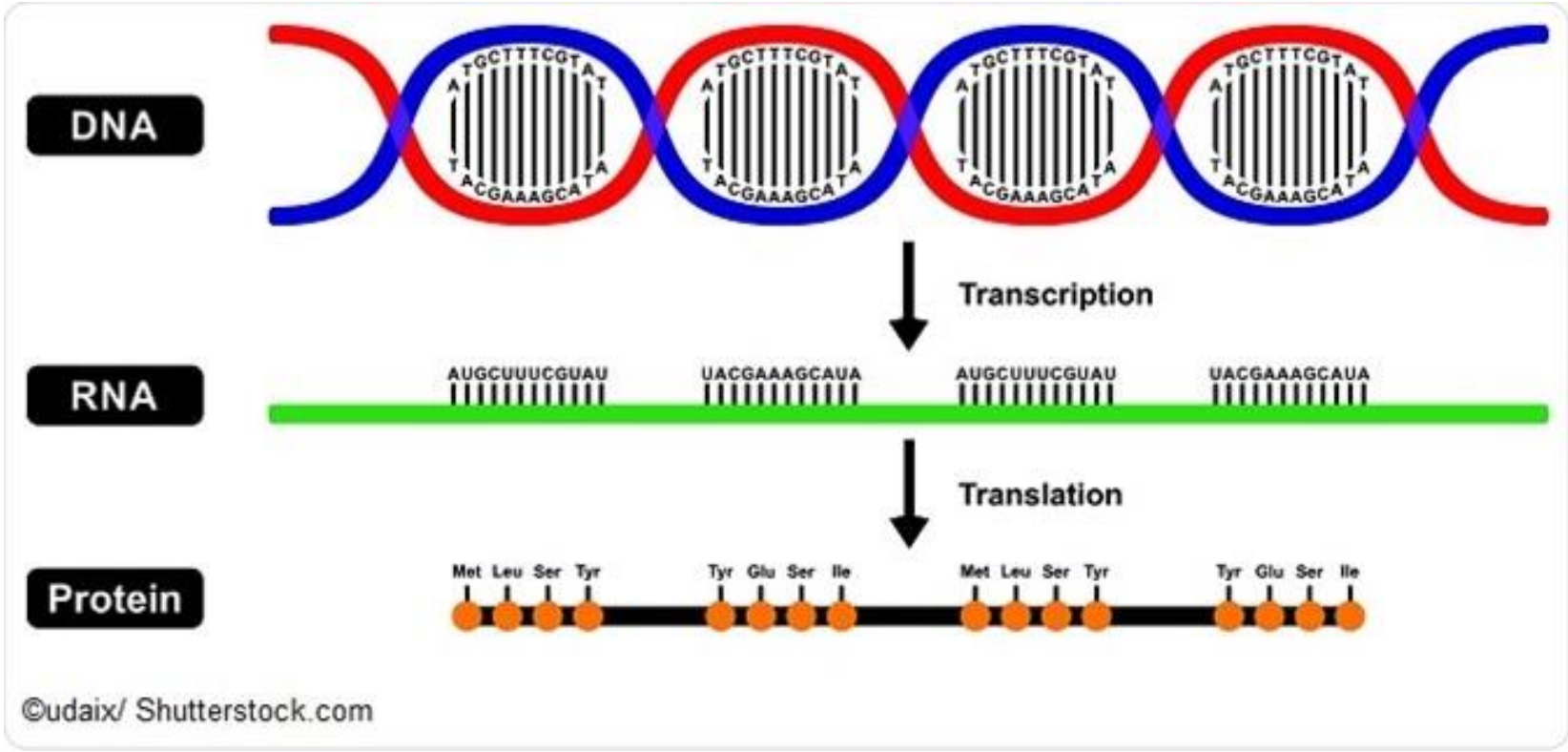


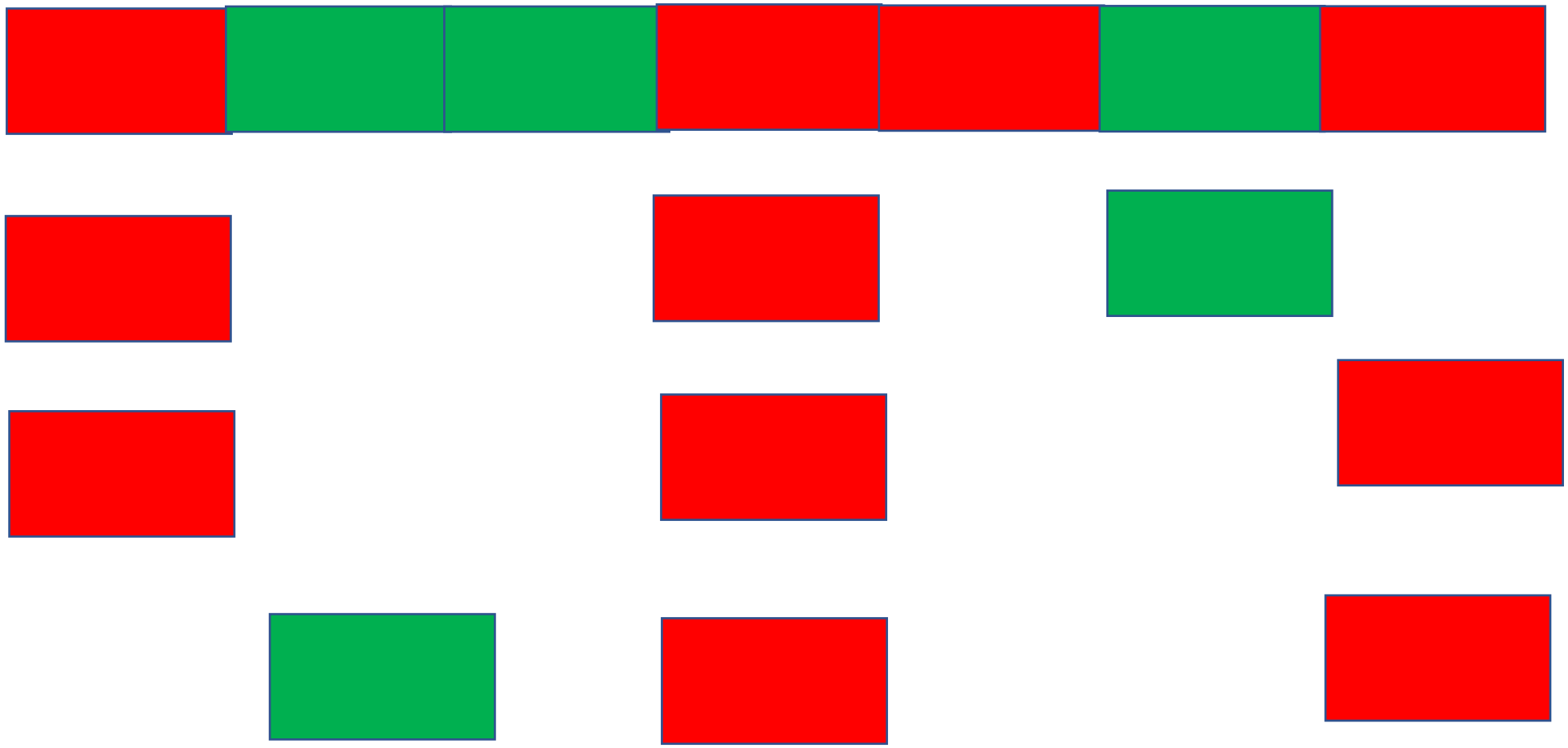
Figure 4: Proportion of extracted sentences by different unsupervised models against their positions

11. 8. Applications to protein and DNA Sequences



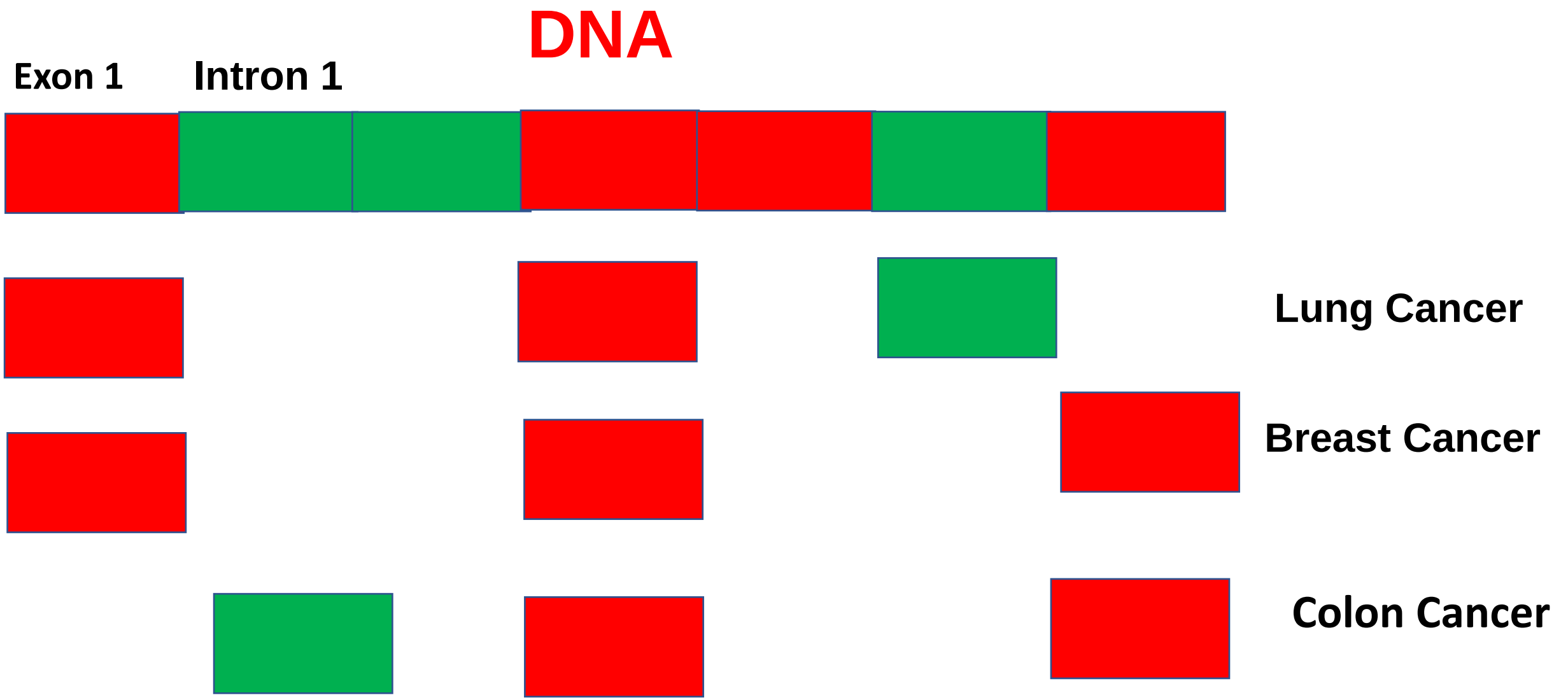
Protein (Gene)

Protein

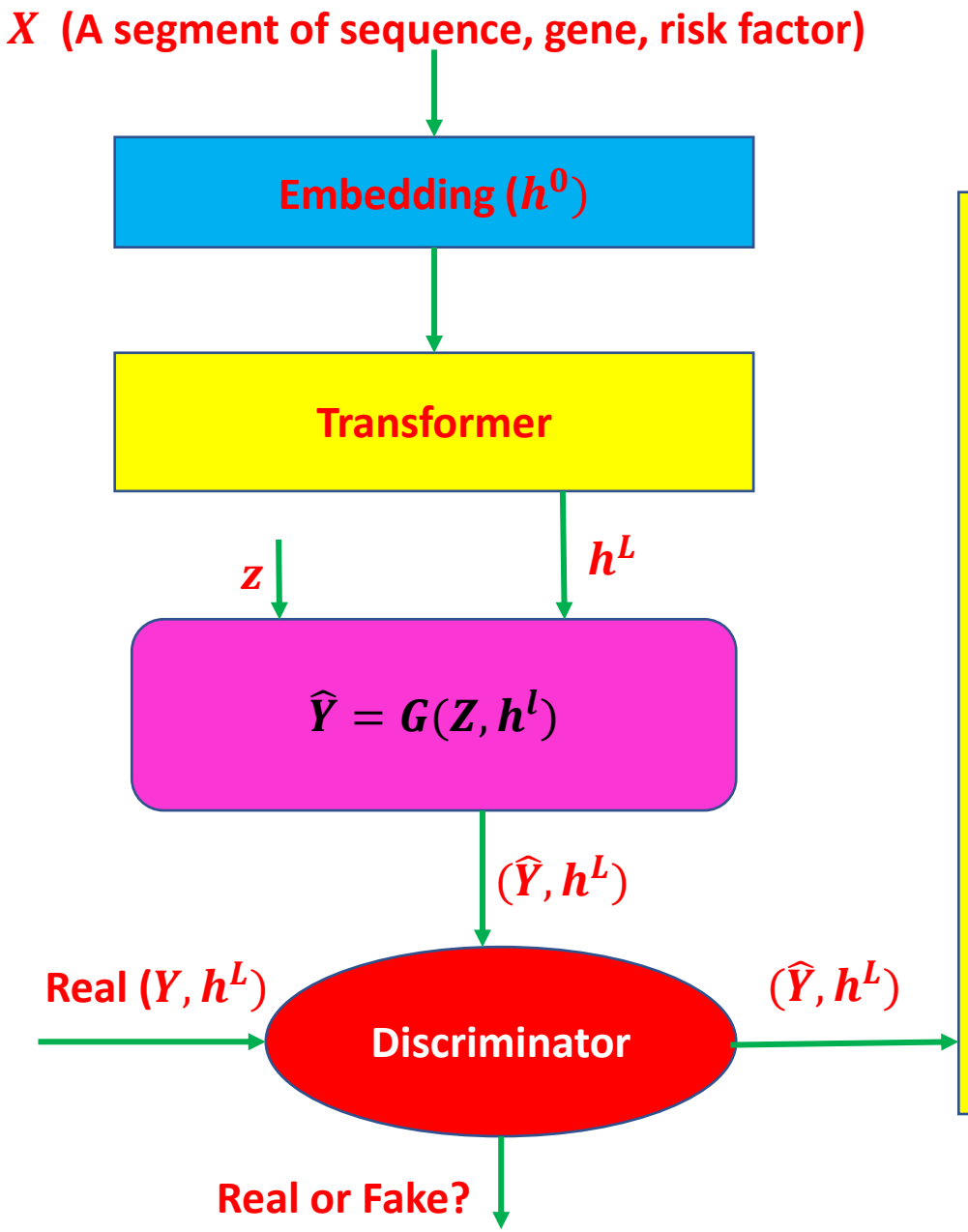


Amino Asid Sequence <-> Word

Protein <-> Sentence


DNA
Exon 1
Intron 1
Lung Cancer
Breast Cancer
Colon Cancer
DNA Sequence <-> Word
Exon - Intro <-> Sentence
Polygenic Score

Pair-wise causal analysis (GAN)



Two Sample Test

$$D_{X\to Y} = \{h_i^L, \hat{Y}_i = G(Z_i, h_i^L), i = 1, \dots, n\}$$

$$D_t = \{h_i^L, Y_i, i = 1, \dots, n\}$$

$$D = \{[(\hat{Y}_1, 1), \dots, (\hat{Y}_n, 1)] \cup [(Y_1, 0), \dots, (Y_n, 0)]\}$$

$$= \{(z_1, l_1), \dots, (z_{2n}, l_{2n})\}$$

$$l = \begin{cases} 1 & \hat{Y}_i \\ 0 & Y_i \end{cases}$$

$$T_{C(X\to Y)} = \frac{1}{n_{te}} \sum_{(h_i, l_i)\in D_{te}} w_i$$

$$w_i = I[I\left(f(h_i) > \frac{1}{2}\right) = l_i]$$

$T_{C(X\to Y)} \sim 0.5, X \to Y$
X causes Y.

Data Sources

How can I get all the proteins involved in a given disease?

https://www.uniprot.org/help/disease_query

Information relevant to diseases associated with a given protein are found in the section 'Disease/Phenotypes and variants'. The information given (including the disease name) is consistent with the literature and the OMIM database.

You can use three methods to search proteins associated with a given disease.

UniProt

https://www.uniprot.org/

DNA Data Sources

Human Genome Resources at NCBI

https://www.ncbi.nlm.nih.gov/genome/guide/human/

OMIM
OMIM is a comprehensive, authoritative compendium of human genes and genetic phenotypes that is freely available and updated daily. OMIM is authored and edited at the McKusick-Nathans Institute of Genetic Medicine, Johns Hopkins University School of Medicine, under the direction of Dr. Ada Hamosh. Its official home is omim.org.

The Human Gene Mutation Database

https://www.hgmd.cf.ac.uk/ac/index.php

Adaptive Skill Coordination for Robotic Mobile Manipulation

Naoki Yokoyama[1], Alexander William Clegg[2], Eric Undersander[2], Sehoon Ha[1], Dhruv Batra[1,2], Akshara Rai[2]
[1]Georgia Institute of Technology, [2]FAIR, Meta AI

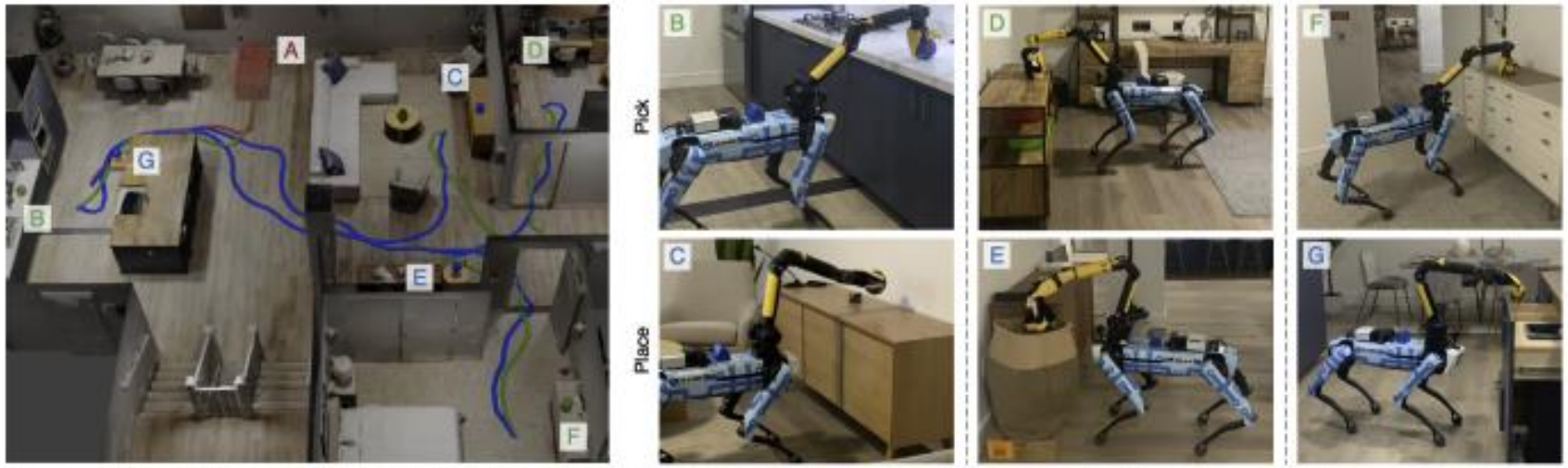
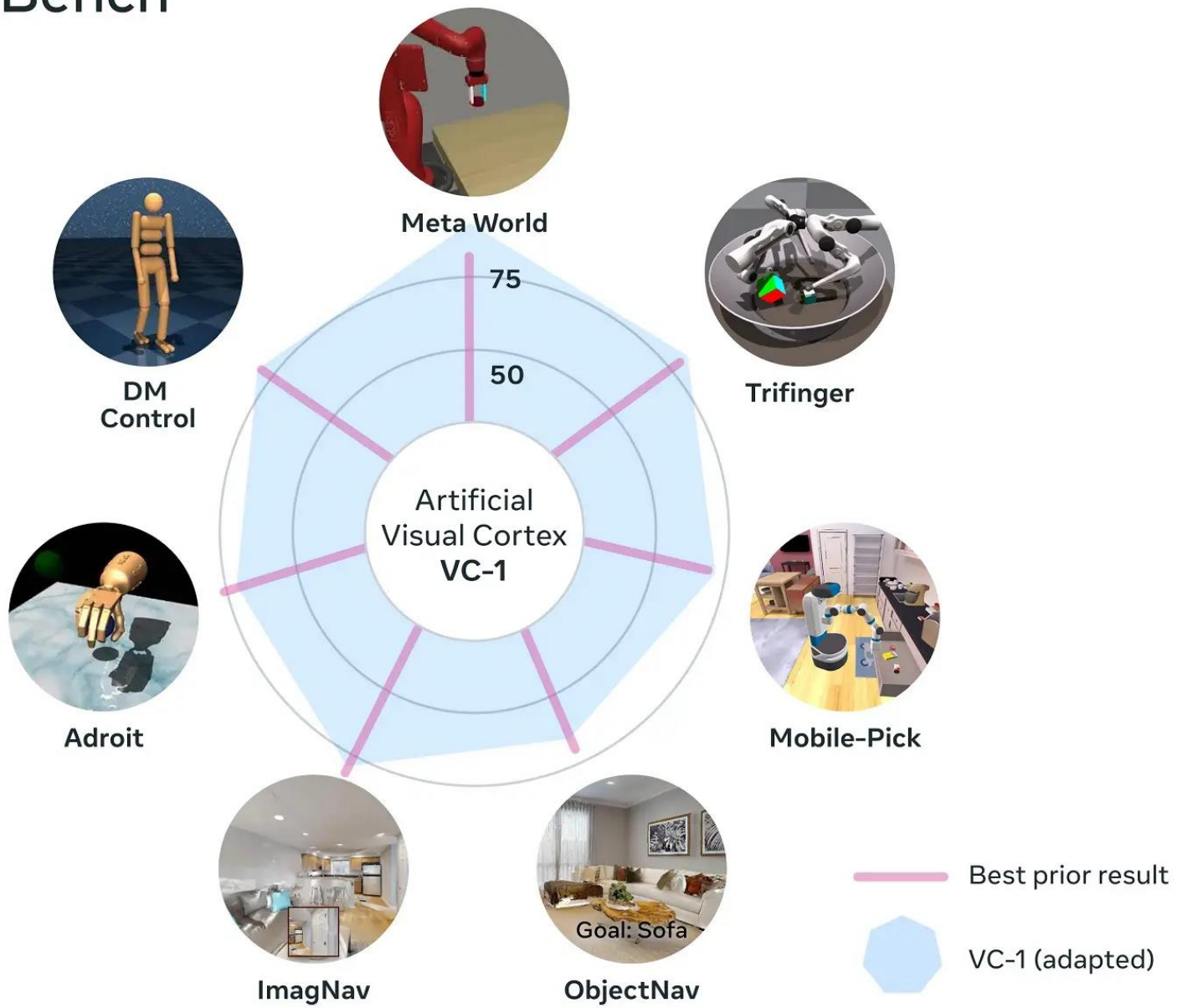


Fig. 1: Adaptive Skill Coordination (ASC) is deployed on Spot in a novel environment and tasked with mobile pick-and-place. ASC operates entirely using onboard sensors – head- and arm-mounted cameras, proprioceptive joint sensors, and egomotion sensors – without access to pre-built maps, 3D models of objects, or precise object locations. Here, Spot navigates from room to room, picking and placing objects, using learned sensor-to-action skills. The robot starts at its dock (red, A), navigates to a pick receptacle (green, B, D, F), searches for and picks an object, navigates to a place receptacle (blue, C, E, G), and places the object at its desired place location, and repeats.

Cortex Bench



References

LONG DOCUMENT SUMMARIZATION WITH TOP-DOWN AND BOTTOM-UP INFERENCE
Bo Pang, Erik Nijkamp, Wojciech Krysci ´ nski, Silvio Savarese, Yingbo Zhou, Caiming Xiong

Investigating Efficiently Extending Transformers for Long Input Summarization
Jason Phang 1∗ Yao Zhao2 Peter J. Liu2

https://github.com/google-research/pegasus/tree/main/pegasus/flax

Discourse-Aware Unsupervised Summarization of Long Scientific Documents
Yue Dong∗

ViT5: Pretrained Text-to-Text Transformer for Vietnamese Language Generation Long
Phan1,2
, Hieu Tran1

Code and models for reproducing our experiments: https://github.com/vietai/ViT5

https://github.com/vietai/vit5

Fine-tune BERT for Extractive Summarization
Yang Liu

The codes to reproduce our results are available at https://github.com/nlpyang/BertSum

Text Summarization with Pretrained Encoders
Yang Liu and Mirella Lapata

BART Text Summarization vs. GPT-3 vs. BERT: An In-Depth Comparison
Karthik Shiraly February 22, 2023
https://www.width.ai/post/bart-text-summarization

COLT5: Faster Long-Range Transformers with Conditional Computation

https://github.com/google/flaxformer

Evolutionary-scale prediction of atomic-level proteinstructure
with a language mode

PEGASUS: Pre-training with Extracted Gap-sentences for Abstractive Summarization

The training code and instructions for using model checkpoints can be found at

https://github.com/google-research/pegasus

Investigating Efficiently Extending Transformers for Long Input Summarization

Jason Phang 1∗ Yao Zhao2 Peter J. Liu2

https://github.com/google-research/pegasus/tree/main/pegasus/flax

Transformers-based Encoder-Decoder Models

UniProt: the Universal Protein Knowledgebase in 2023
The UniProt Consortium
Nucleic Acids Research, Volume 51, Issue D1, 6 January 2023, Pages D523–D531,
https://doi.org/10.1093/nar/gkac1052

The Nucleotide Transformer: Building and Evaluating Robust Foundation Models for Human Genomics

Species-aware DNA language modeling

Evolutionary-scale prediction of atomic-level protein structure with a language mode

Representation of missense variants for predicting modes of action