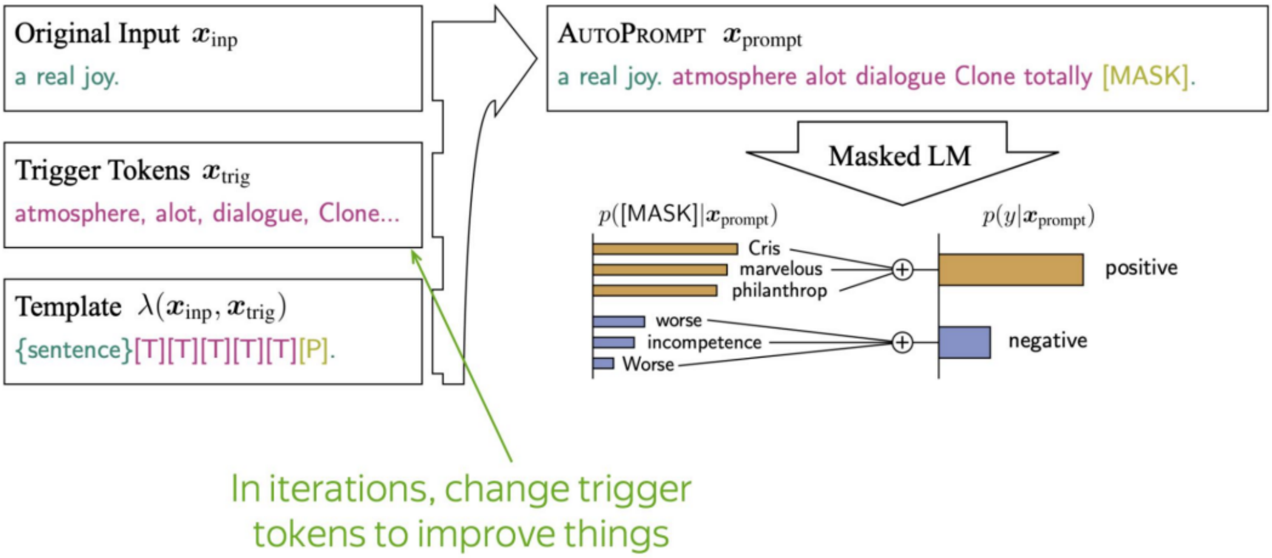
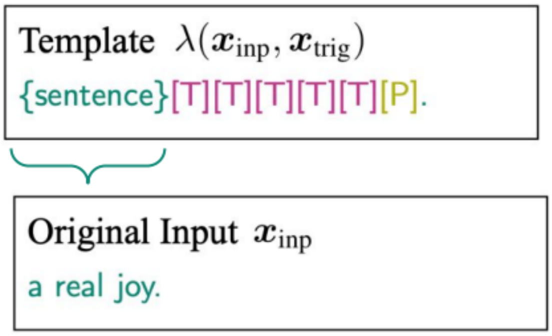
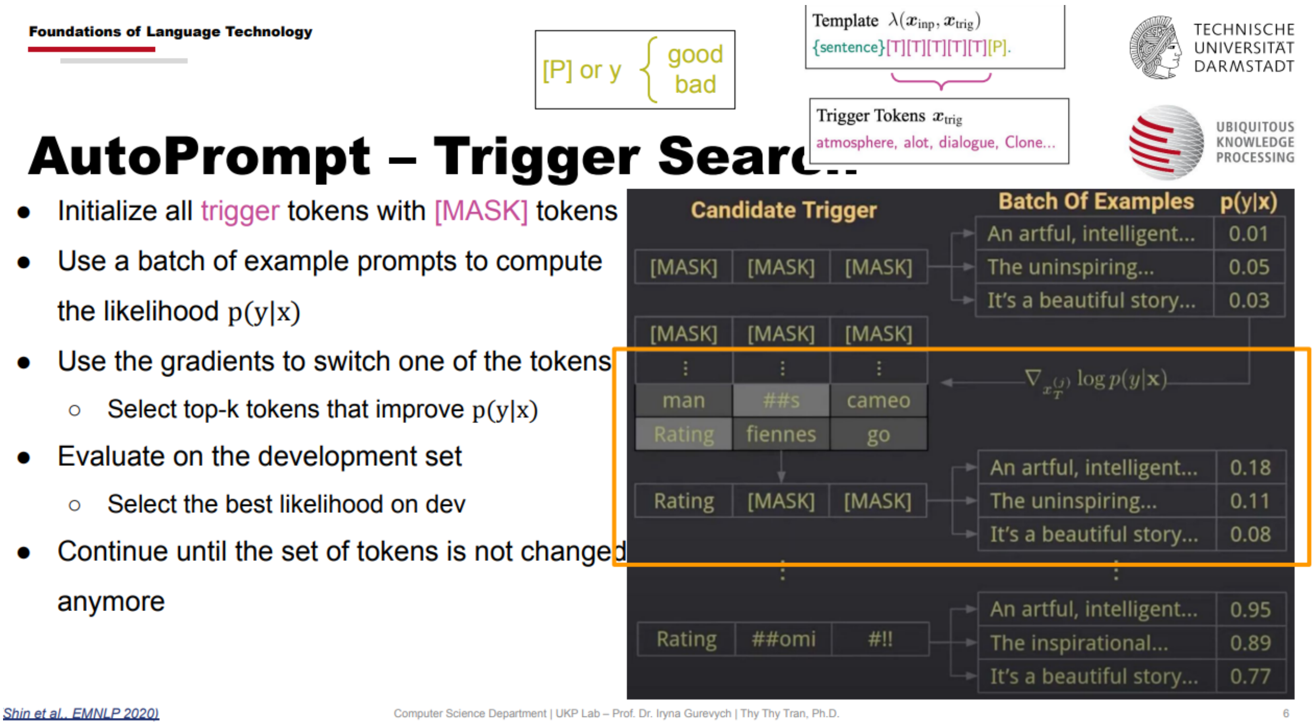
FOLT Lecture 7, Text Generation

Intro : AutoPrompt recap



AutoPrompt trigger search





AutoPrompt – Label Token Search

Template $\lambda(\boldsymbol{x}_{\mathrm{inp}}, \boldsymbol{x}_{\mathrm{trig}})$

{sentence}[T][T][T][T][T][P].

[MASK].

Predict token

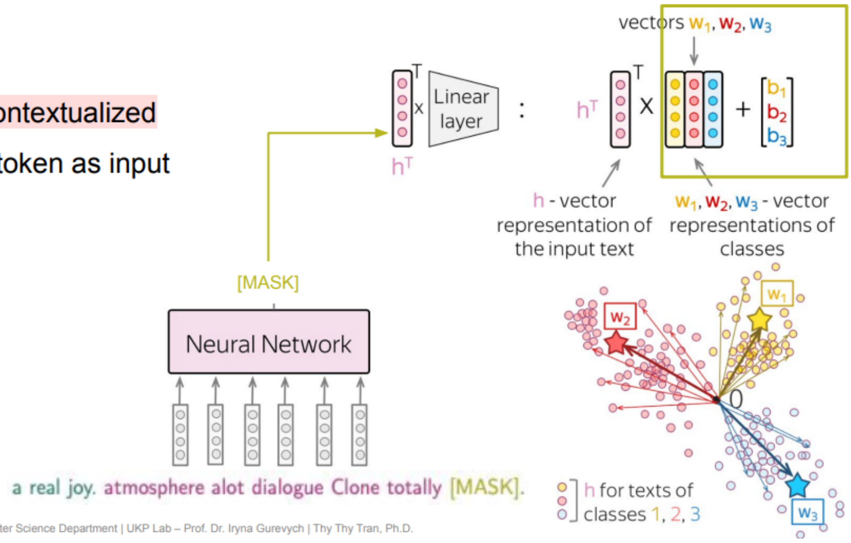
Template $\lambda(\boldsymbol{x}_{\mathrm{inp}}, \boldsymbol{x}_{\mathrm{trig}})$
{sentence}[T][T][T][T][T][P].

Predict token [MASK]

AutoPrompt – Label Token Search

Two-step approach

1. Train a classifier using the contextualized embedding h of the [MASK] token as input



Template $\lambda(\boldsymbol{x}_{\mathrm{inp}}, \boldsymbol{x}_{\mathrm{trig}})$
{sentence}[T][T][T][T][T][P].

Predict token [MASK]

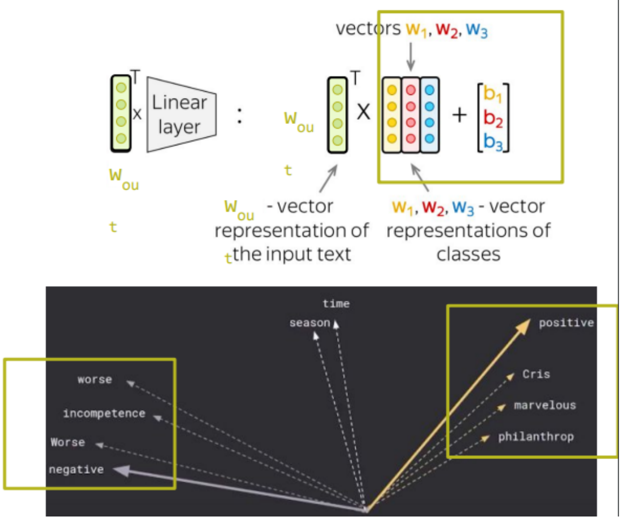
TECHNISCHE UNIVERSITÄT DARMSTADT
UBIQUITOUS KNOWLEDGE PROCESSING

AutoPrompt – Label Token Search

Two-step approach

1. Train a classifier using the contextualized embedding h of the [MASK] token as input

2. Substitute h with LM output word embeddings w_{out}
3. Select the top-k w_{out} with highest scores



Overview

Text gen examples : Machine Translation, Copilot, Text Summarization, Dialogue systems ....

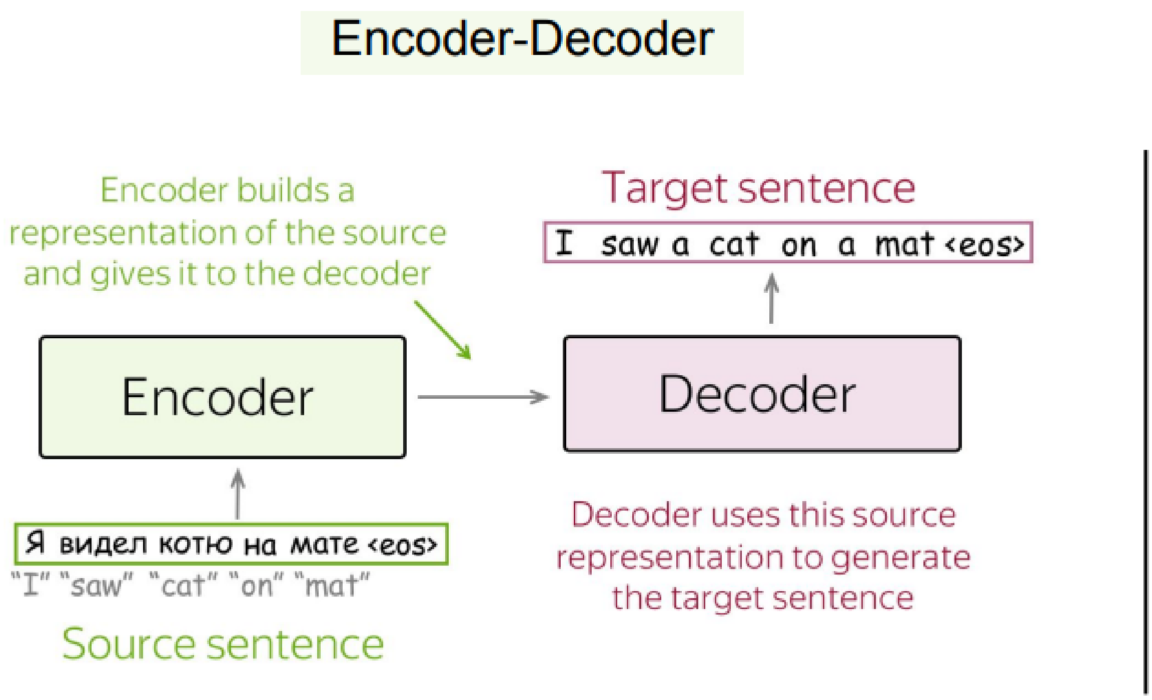
- A text sequence → text sequence in **similar length**
  - E.g., Machine Translation
- A text sequence → text sequence in **much shorter length**
  - E.g., Summarization
- A text sequence → text sequence in **varying lengths**
  - E.g., Conversation/Dialogue

Key Concepts from previous lectures:

- Language models from lecture 2
- Neural language models lecture 3(Predict next word(token))
- Text generation with neural LM (lecture 2)

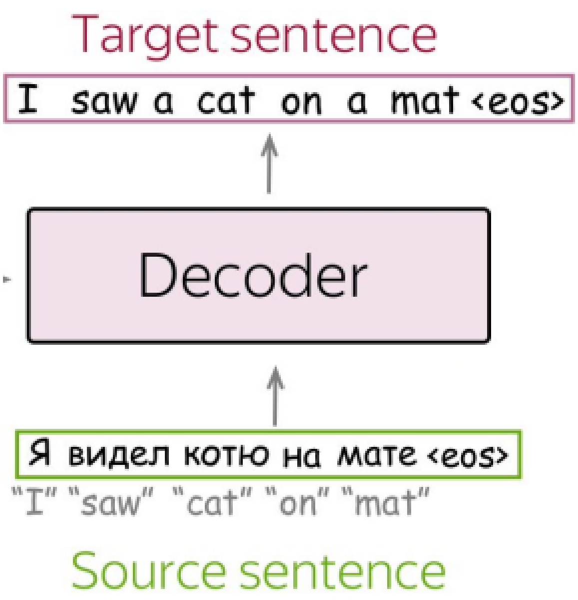
PART1: Generative models

Examples :



Encoder : processes the source sequence and produces its representation(s)
Decoder : uses the source representation(s) from the encoder to generate the target sequence

Decoder-only



Also there is Encoder-only (e.g., BERT) but not covered in this lecture

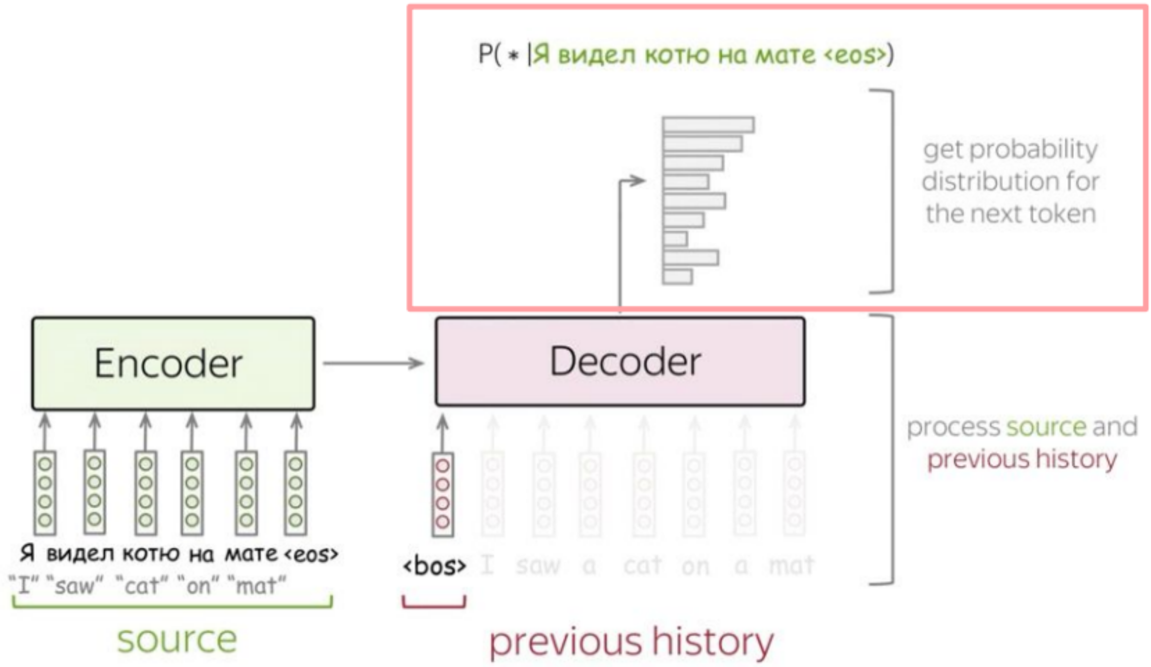
1) Conditional Language Models

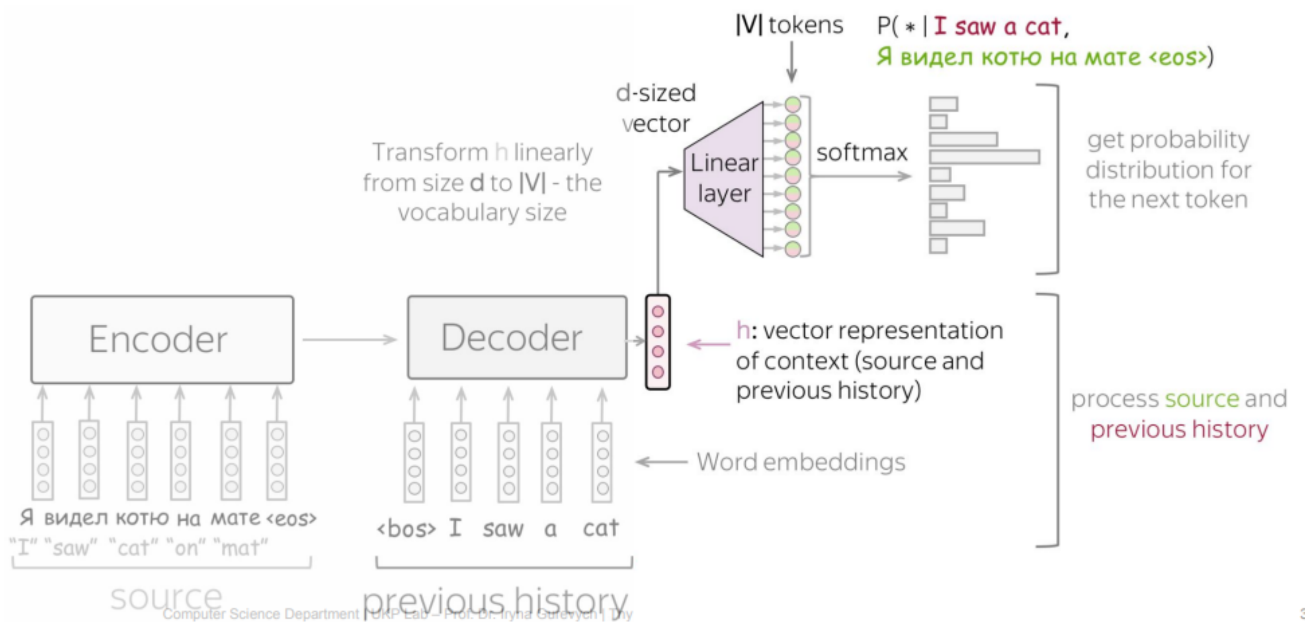
Language Models: (left-to-right) $P(y_1, y_2, \ldots, y_n) = \prod_{t=1}^{n} p(y_t | y_{<t})$

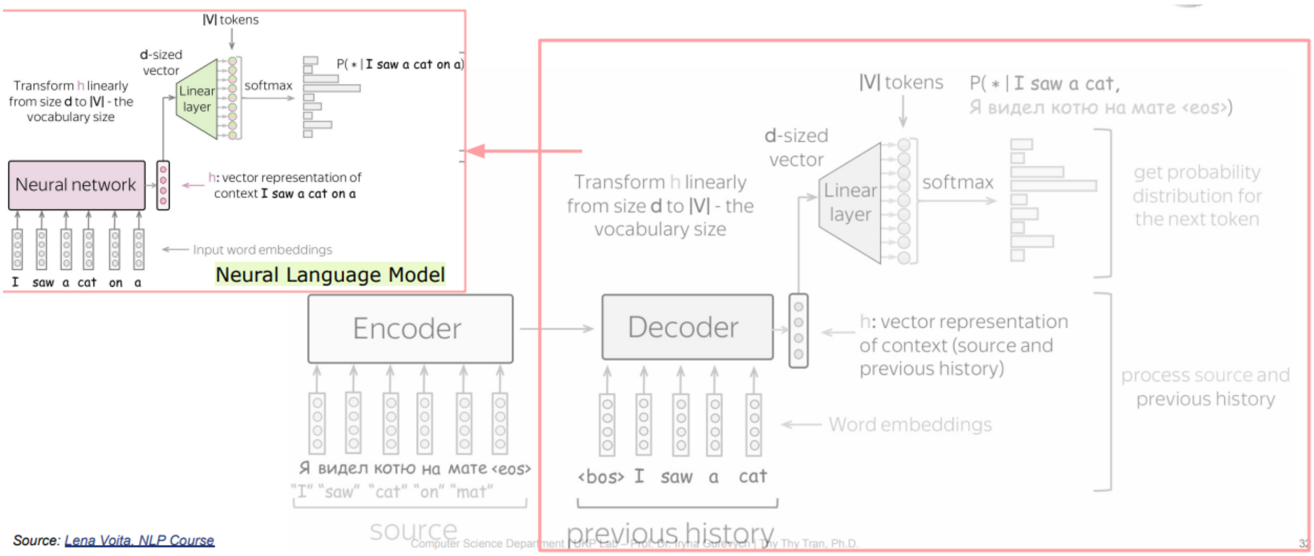
Conditional Language Models: $P(y_1, y_2, \ldots, y_n, | x) = \prod_{t=1}^{n} p(y_t | y_{<t}, x)$

condition on source x

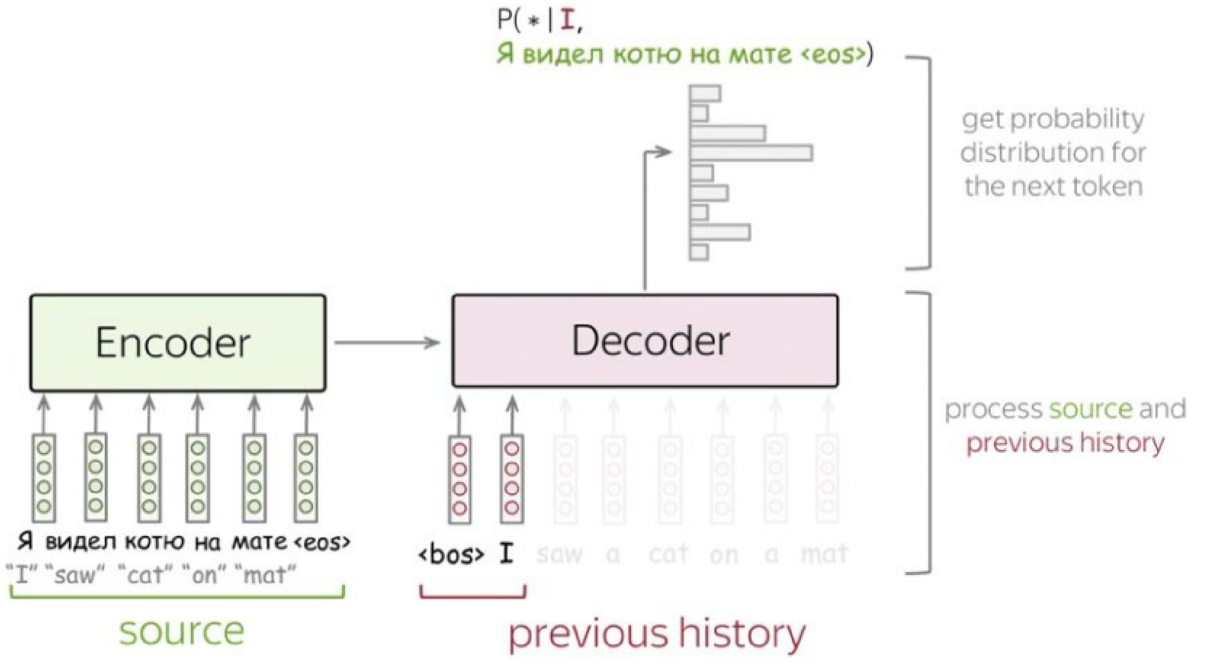
2) Generation with Encoder-Decoder

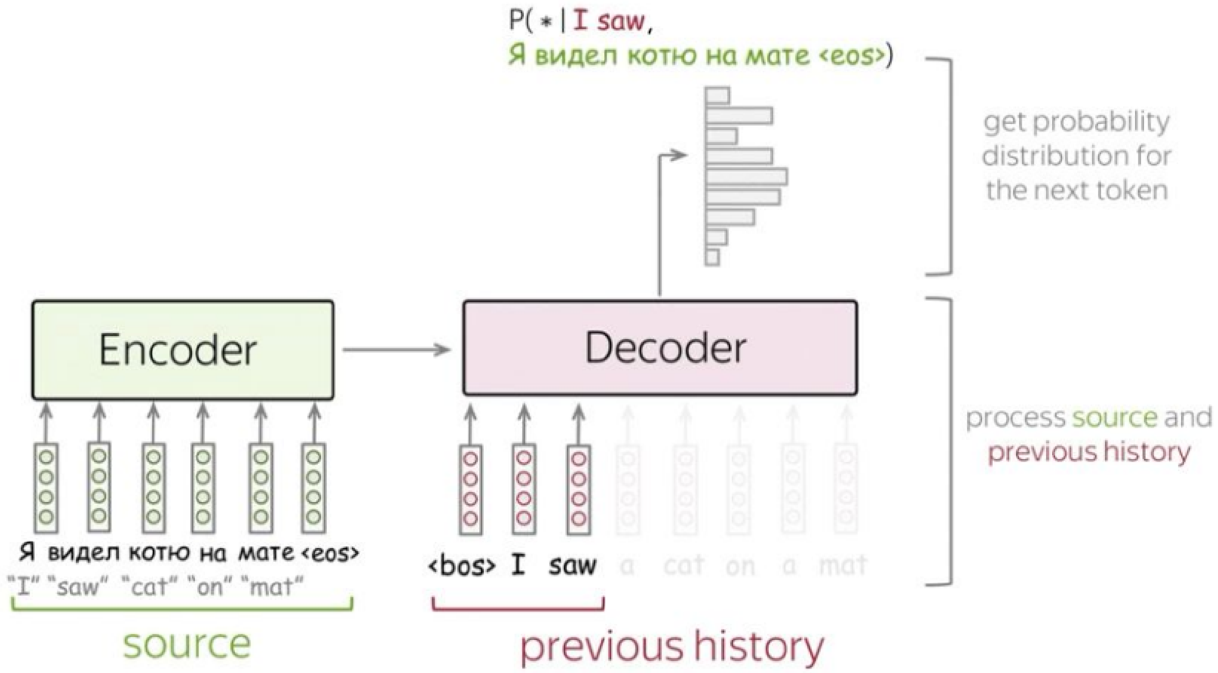


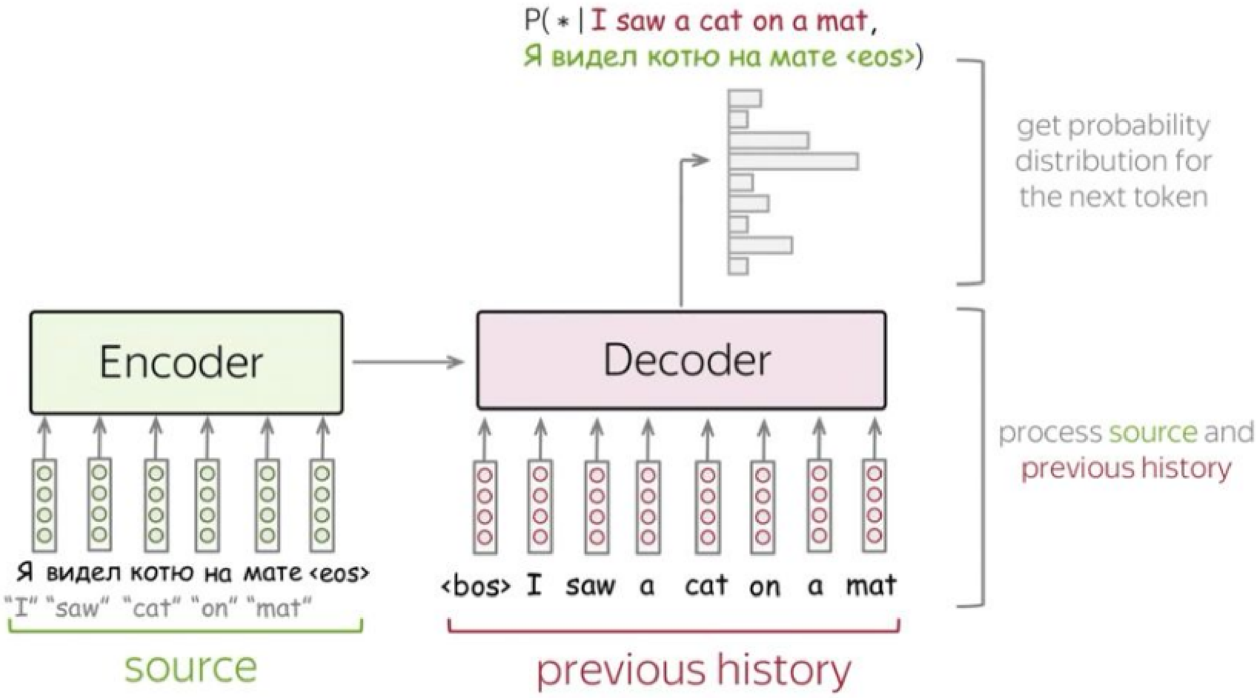


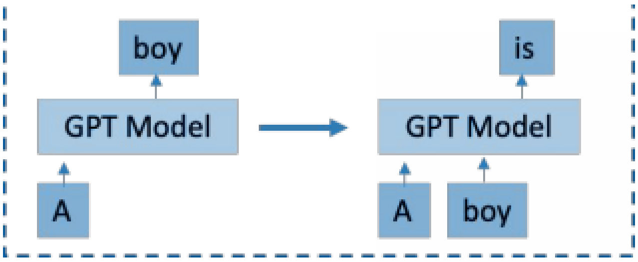


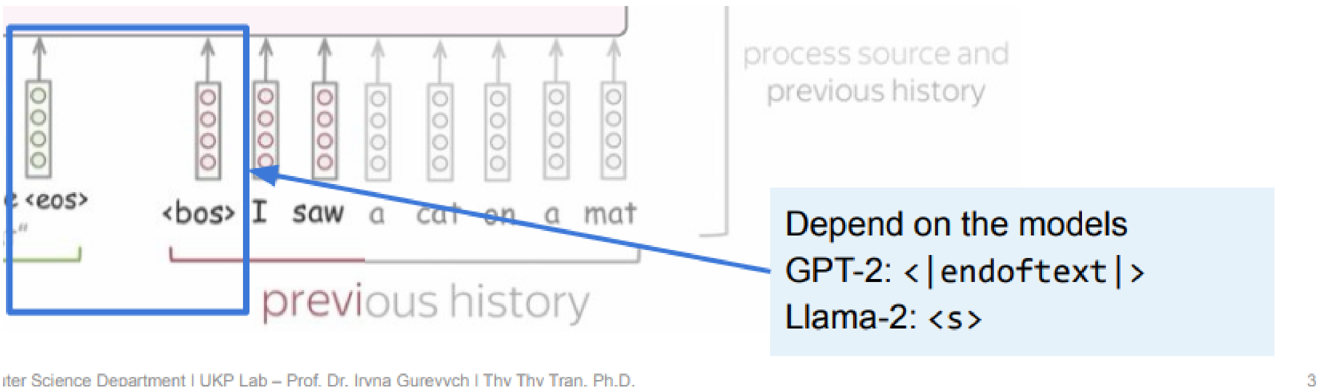
Source: Lena Voita, NLP Course


P(* | I,
Я видел котю на мате <eos>)
get probability distribution for the next token
Encoder
Decoder
process source and previous history
Я видел котю на мате <eos>
"I" "saw" "cat" "on" "mat"
source
<bos> I saw a cat on a mat
previous history


P(* | I saw,
Я видел котю на мате <eos>)
get probability distribution for the next token
Encoder
Decoder
process source and previous history
Я видел котю на мате <eos>
"I" "saw" "cat" "on" "mat"
source
<bos> I saw a cat on a mat
previous history

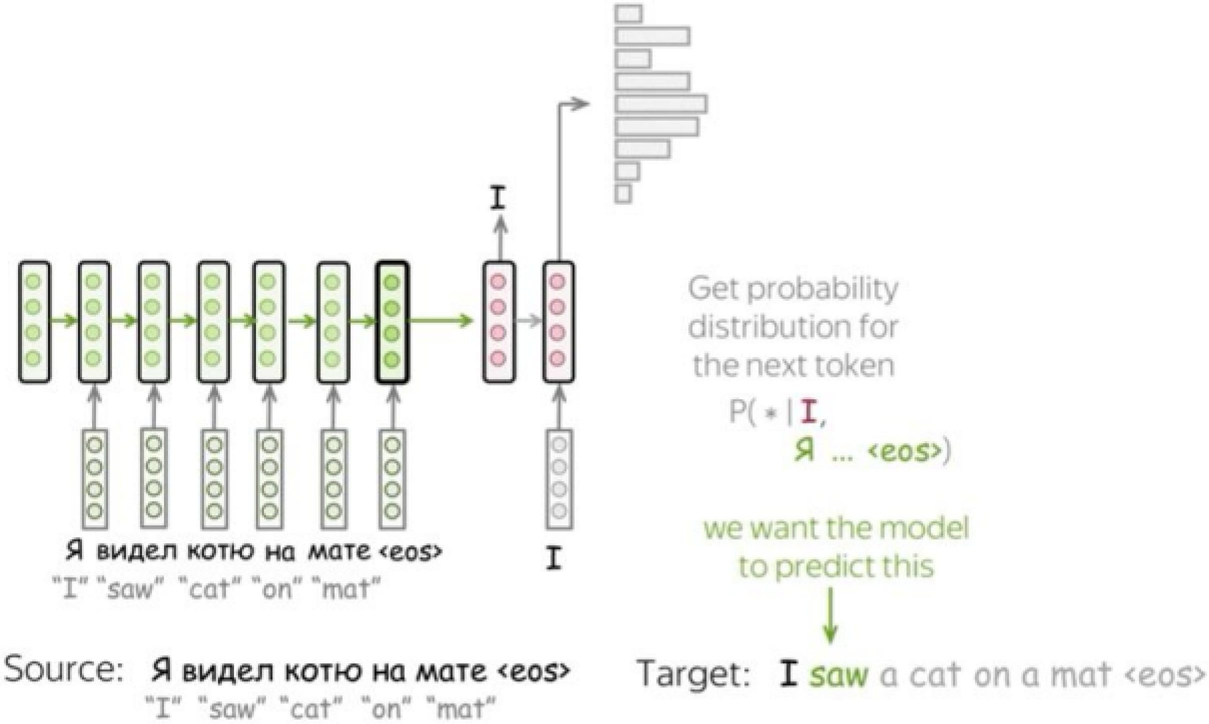


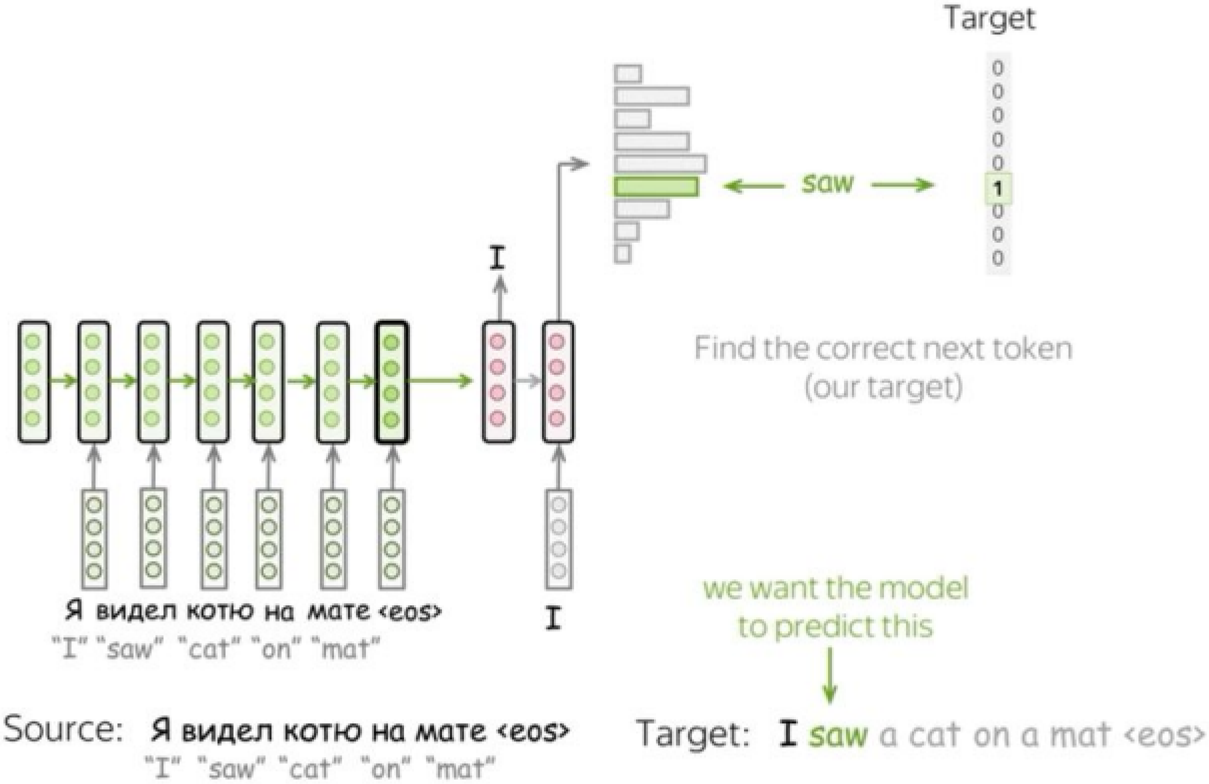


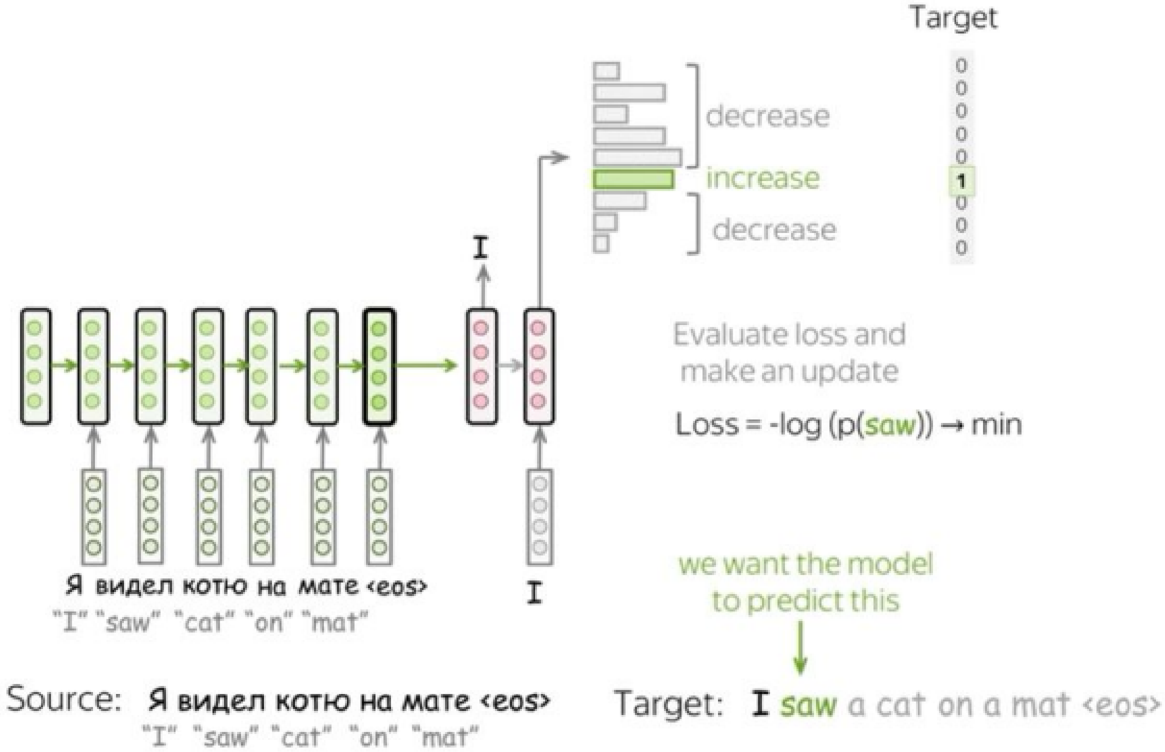


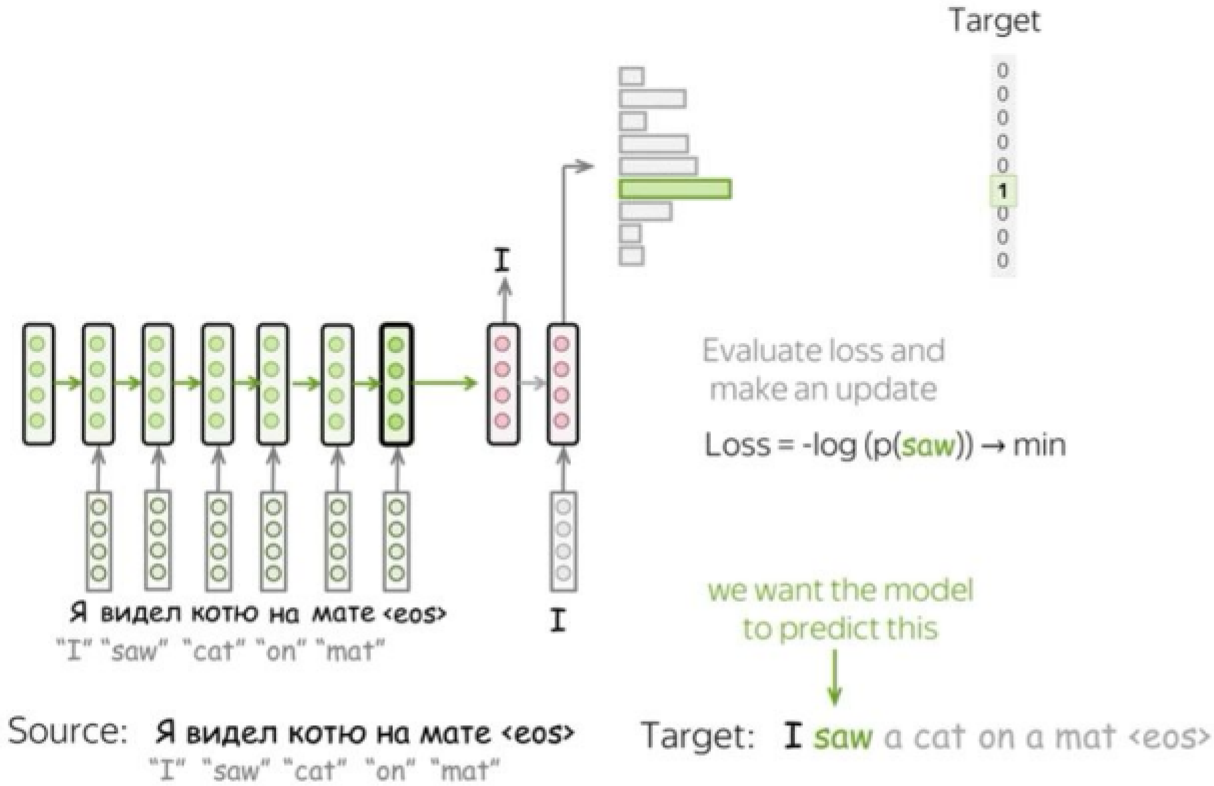
3) Training text generation

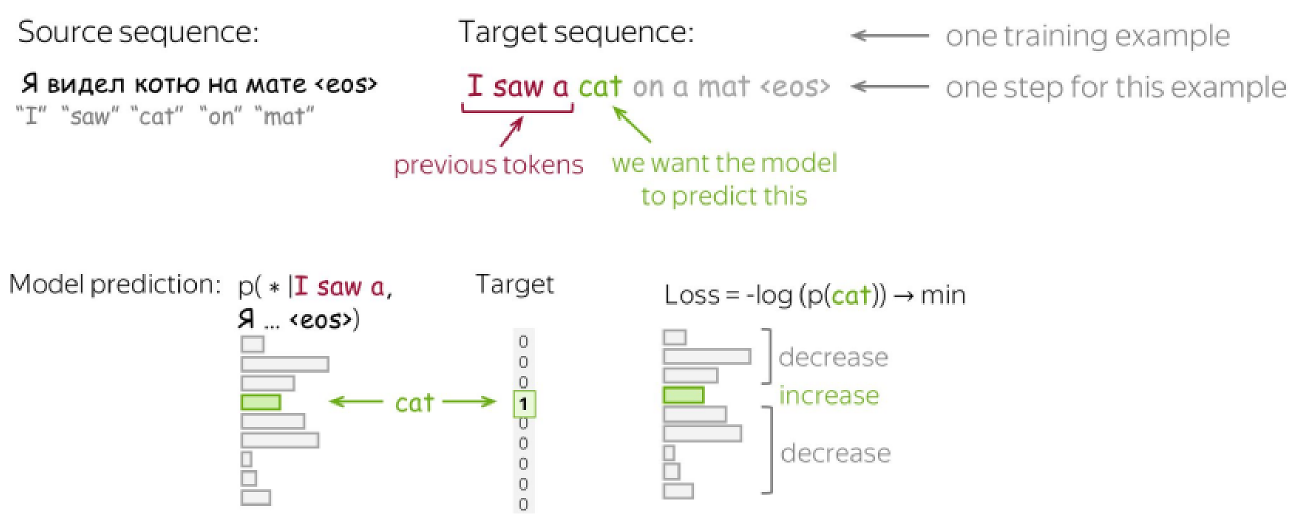
Similar to neural LMs, neural text generation models are trained to maximize the probability distribution of the next token given the previous context


I
Get probability distribution for the next token
P(* | I,
Я ... <eos>)
Я видел котю на мате <eos>
"I" "saw" "cat" "on" "mat"
I
we want the model to predict this
Source: Я видел котю на мате <eos>
"I" "saw" "cat" "on" "mat"
Target: I saw a cat on a mat <eos>


Target
0
0
0
0
0
1
0
0
0
saw
I
Find the correct next token (our target)
Я видел котю на мате <eos>
"I" "saw" "cat" "on" "mat"
I
we want the model to predict this
Source: Я видел котю на мате <eos>
"I" "saw" "cat" "on" "mat"
Target: I saw a cat on a mat <eos>


Target
0
0
0
0
0
1
0
0
0
decrease
increase
decrease
I
Evaluate loss and
make an update
Loss = -log (p(saw)) → min
Я видел котю на мате <eos>
"I" "saw" "cat" "on" "mat"
I
we want the model
to predict this
Source: Я видел котю на мате <eos>
"I" "saw" "cat" "on" "mat"
Target: I saw a cat on a mat <eos>

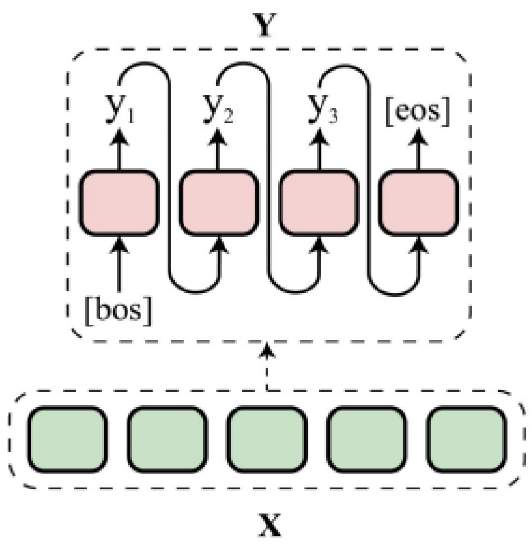

Target
0
0
0
0
0
1
0
0
0
I
Evaluate loss and
make an update
Loss = -log (p(saw)) → min
Я видел котю на мате <eos>
"I" "saw" "cat" "on" "mat"
I
we want the model
to predict this
Source: Я видел котю на мате <eos>
"I" "saw" "cat" "on" "mat"
Target: I saw a cat on a mat <eos>



PART 2: Decoding Strategies(Generating text)

1)Autoregressive Generation

Similar to text generation with neural LMs



Autoregressive Generation [AG]

- Starts with [bos] (begin-of-sequence)
- At each step
  - process previous generated tokens
  - get probability distribution for the next token
- Stops by [eos] (end-of-sequence)
  - Terminate when [eos] is predicted
  - Or stop generating text when max target sequence length is reached
    - max target sequence length or max new tokens: expected maximum length of Y
    - max token length: expected maximum length of X + Y

2) Probability for Next Token

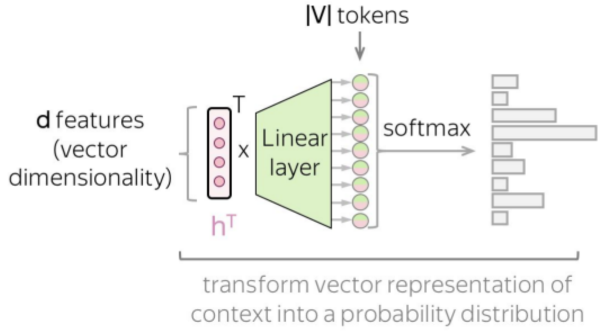
We have:

- h - vector of size d

We need:

- vector of size |V| – probabilities for all tokens in the vocabulary

Transform linearly from size d to size |V|



Conditional Language Models: $P(y_1, y_2, \dots, y_n, | x) = \prod_{t=1}^{n} p(y_t | y_{<t}, x)$

condition on source x

$$\mathbf{y} = \mathrm{softmax}(\mathbf{hW} + \mathbf{b}) \rightarrow \frac{\boxed{\exp(\mathbf{z}_c)}}{\sum_{c' \in C} \exp(\mathbf{z}_{c'})}$$

$\boldsymbol{h} \in \mathbb{R}^d$, $\mathbf{W} \in \mathbb{R}^{d \times |V|}$, $\mathbf{b} \in \mathbb{R}^{|V|}$

$$p(y_t = w \mid y_{<t}, x) \propto \boxed{\exp(\mathrm{score}(w))}$$

3) Temperature – Tempered Sampling

$$p(y_t = w \mid y_{<t}, x) \propto \exp(\mathrm{score}(w))$$

$$\boxed{q(y_t = w \mid y_{<t}, x) \propto \exp(\mathrm{score}(w)/T) \quad \text{where } T \in (0, +\infty)}$$

- Typically we chooose $T \in (0, 1)$, which makes the distribution more peaky.

What happens when $T \rightarrow 0$ and $T \rightarrow +\infty$?

4)Repetition Penalty

[Keskar et al., 2019] Intuition: to avoid generating duplicate substrings

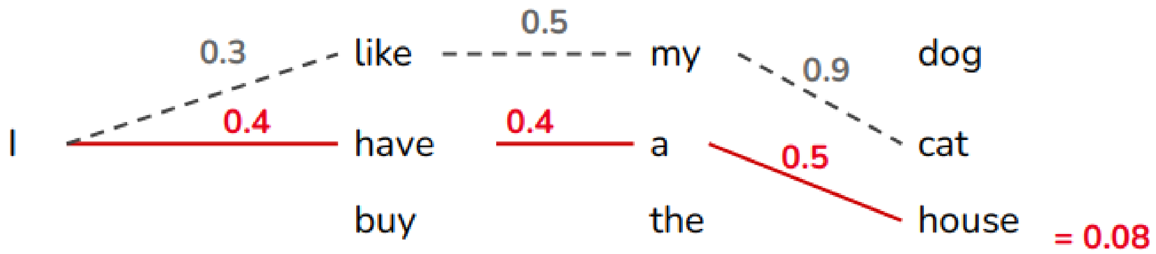
$$p_i = \frac{\exp(x_i/(T \cdot I(i \in g))}{\sum_j \exp(x_j/(T \cdot I(j \in g))} \qquad I(c) = \theta \text{ if c is True else } 1$$

- g contains a set of previously generated tokens
- $\mathbb{1}$ is an identity function
- θ = 1.2 is found to yield a good balance between less repetition and truthful generation.

5) Decoding Strategies

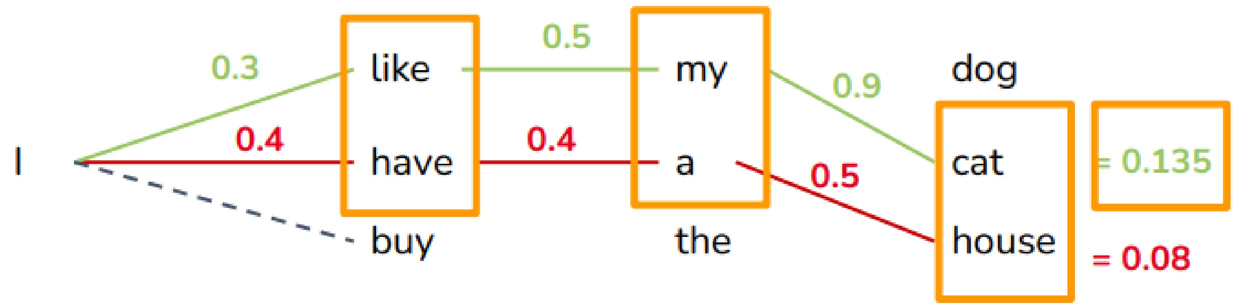
	Greedy	Beam Search	Top-k Sampling	Top-p (Nucleus) Sampling
At each step	Pick the best word	Try a few best words	Random sample from top-k	smallest set with cumulative probability > p
Output	One sequence	Several partial sequences	One sequence	One sequence

a) Greedy Decoding



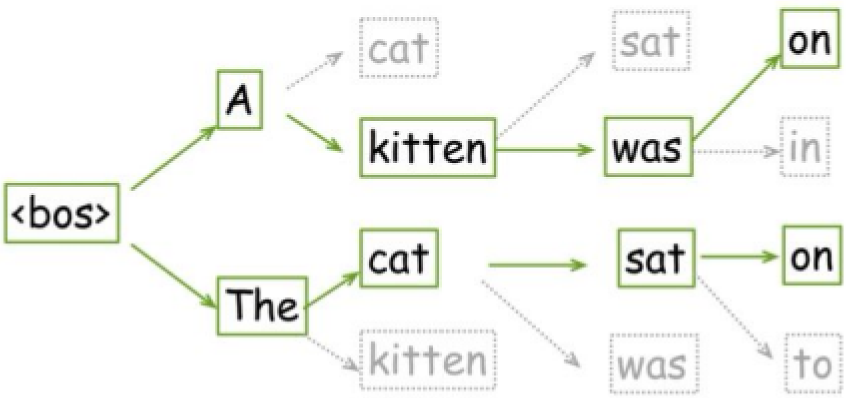
Weakness : Repetition as it always selects the most frequent token

b) Beam Search



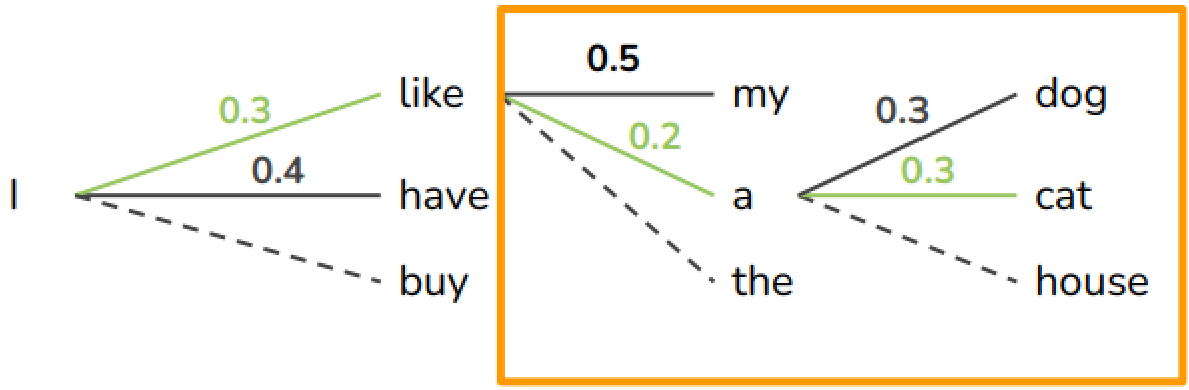
● Top-N is called beam size, usually 4-10.
● Increasing beam size is computationally inefficient and may lead to worse quality.
Strategy:
Start with the beginning of the sentence token or with an empty sequence

Pick top **beam_size** hypos ,terminate the test



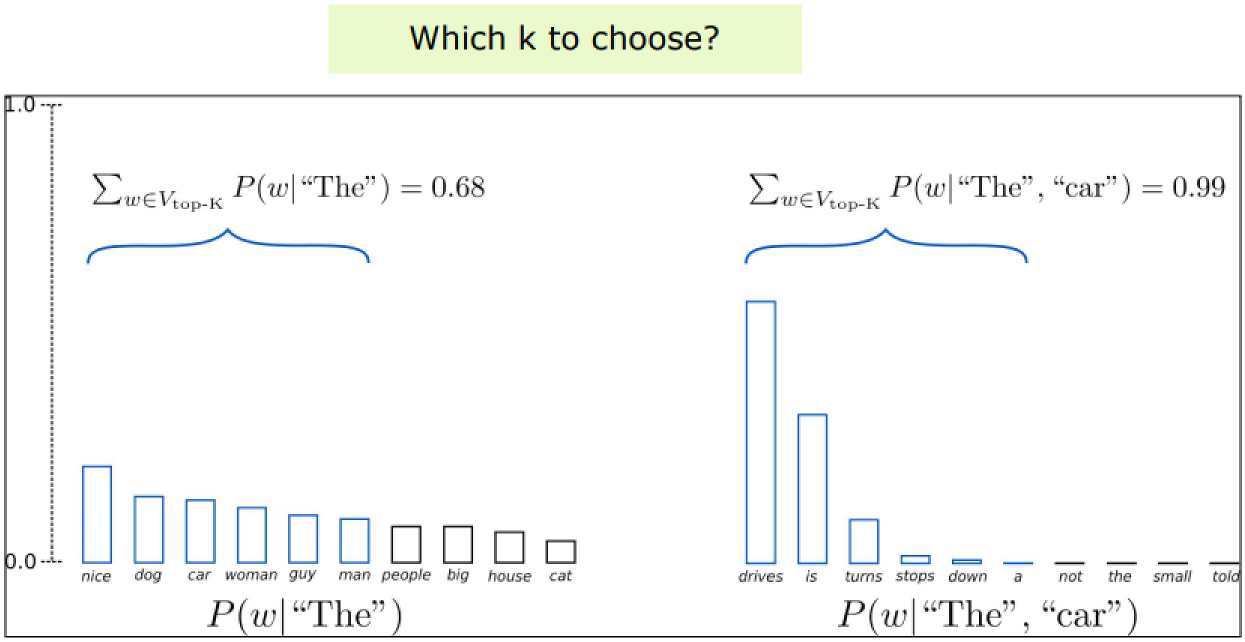
Weakness:
● Short sentences
● Less diversity
Beam search Text is not surprising

c)Top-k Sampling



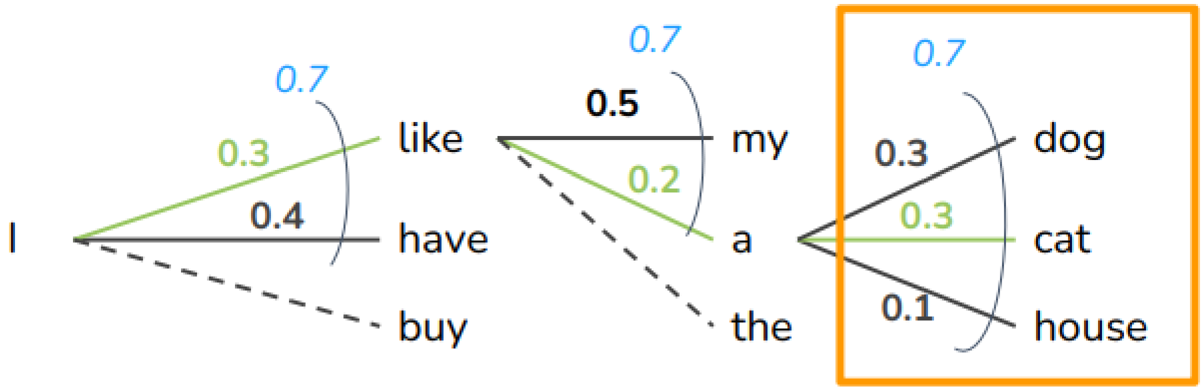
- Sort the probabilities of the vocabulary at each step
- Select the top-k words, k is often 5 – 20
- k=1 ⇒ greedy decoding
- Increase k → have more diverse, also more risky
- Decrease k → have more safe choices but less diverse

Which k to choose?



Weakness ● Weird n-grams may occur due to random picking of top-k words ⇒ the output may not be coherent

d)Top-p (Nucleus) Sampling



● Commonly-used probability p is 0.95

Weakness ● (Surprisingly) may not include surprise words

e) Decoding in practise:

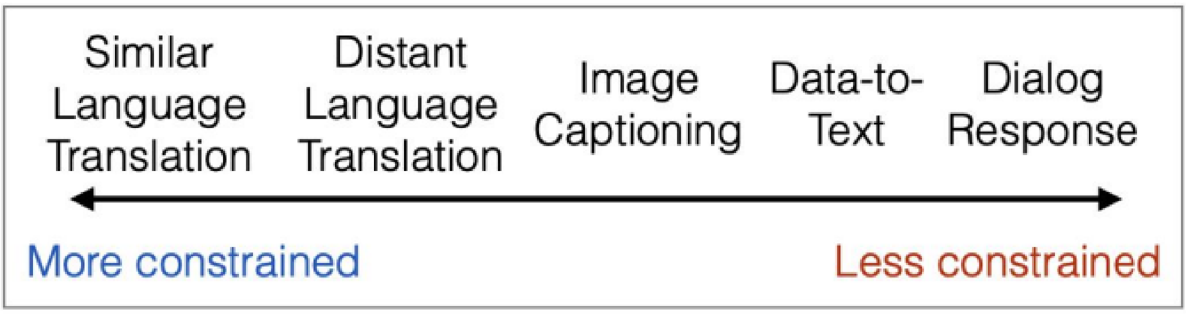
● Can combine different strategies

○ e.g., temperature + beam search, temperature + top-k

● Use beam search with small beam size for tasks that exists a correct answer (more constrained)

● Use top-k or top-p for open-ended generation (less constrained)

● As models getting better/larger, sampling-based methods tend to work better



More freedom = more flexibility, but often more difficulty in modeling and evaluation

f) Controlled Generation

● Add a further constraint in addition to content-based ones

● Politeness/Style Control: Take an input X and a label indicating style, etc.

source	Give me the telephone!
reference	Gib mir das Telefon! [T]
none	Gib mir das Telefon! [T]
polite	Geben Sie mir das Telefon! [V]
informal	Gib mir das Telefon! [T]

● Personalization: Take an input X and a side information about the speaker

English Sentence: Accordingly , I consider it essential that both the identification of cattle and the labelling of beef be introduced as quickly as possible on a compulsory basis .

German Sentence: Entsprechend halte ich es auch für notwendig , daß die Kennzeichnung möglichst schnell und verpflichtend eingeführt wird , und zwar für Rinder und für Rindfleisch .

Meta Info: EUROID="2209" NAME="Schierhuber" LANGUAGE="DE" GENDER="FEMALE" DATE_OF_BIRTH="31 May 1946" SESSION_DATE="97-02-19" AGE="50"