PROPOR 2020 - Évora, Portugal

Modelos de Linguagem

Joaquim Santos, Renata Vieira

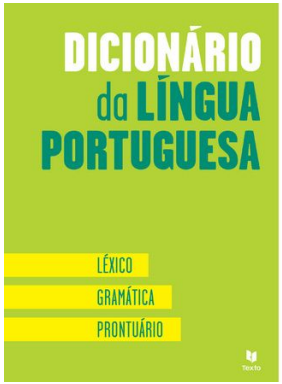
Roteiro

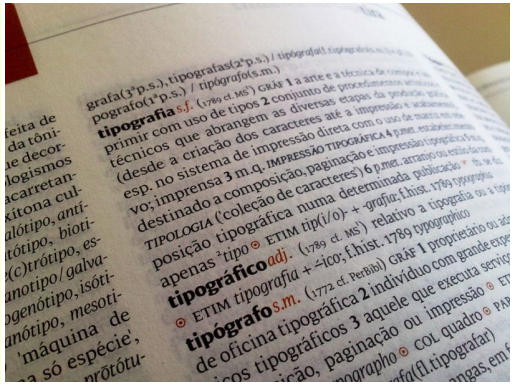
1. Introdução
2. Fundamentos matemáticos
3. Histórico
4. Modelos WE iniciais
5. Modelos dinâmicos, sensíveis ao contexto
6. Modelos para a língua portuguesa
7. Modelos específicos de domínio
8. Geração de modelos
9. Aplicações
10. Avaliação de modelos
11. Limitações

Introdução

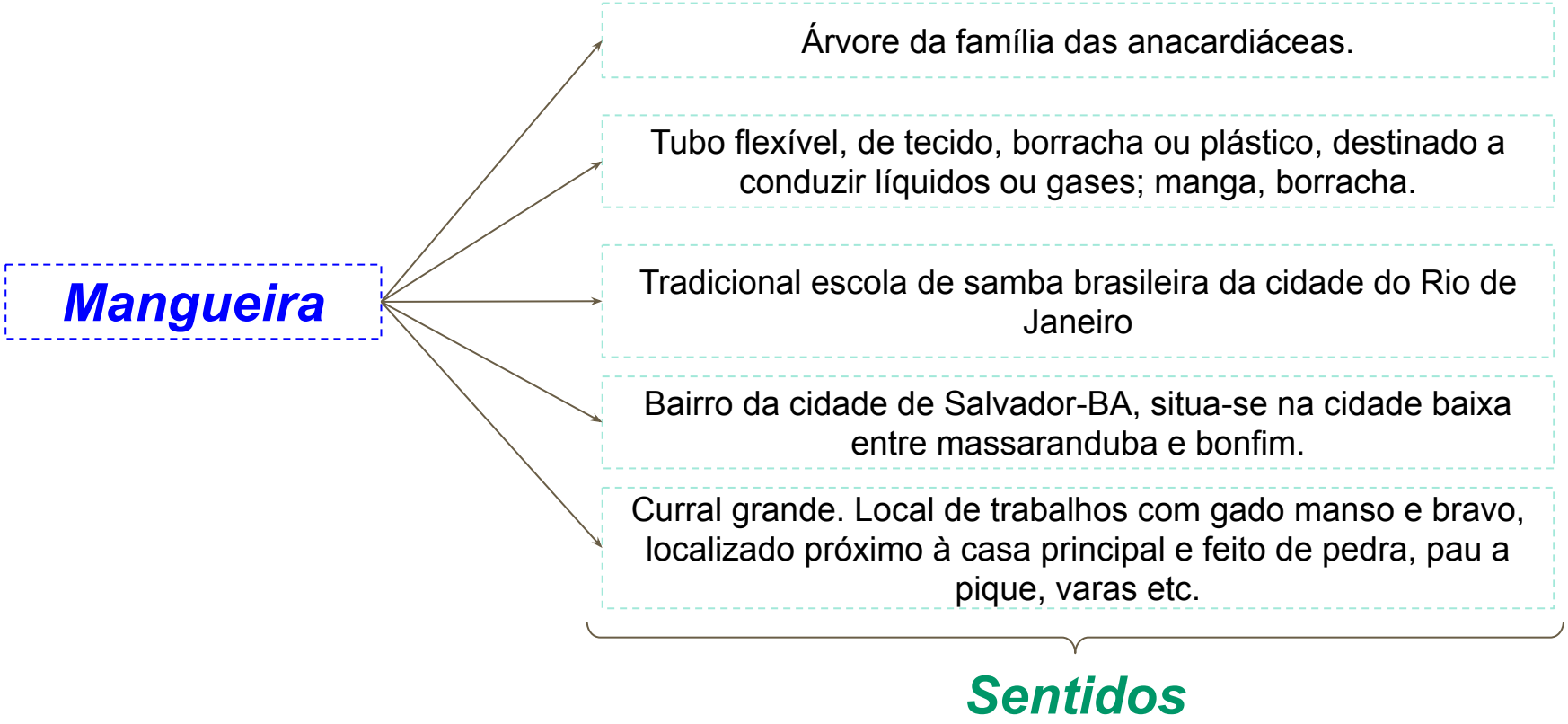
LÍNGUA E SIGNIFICADO

O que as palavras significam?

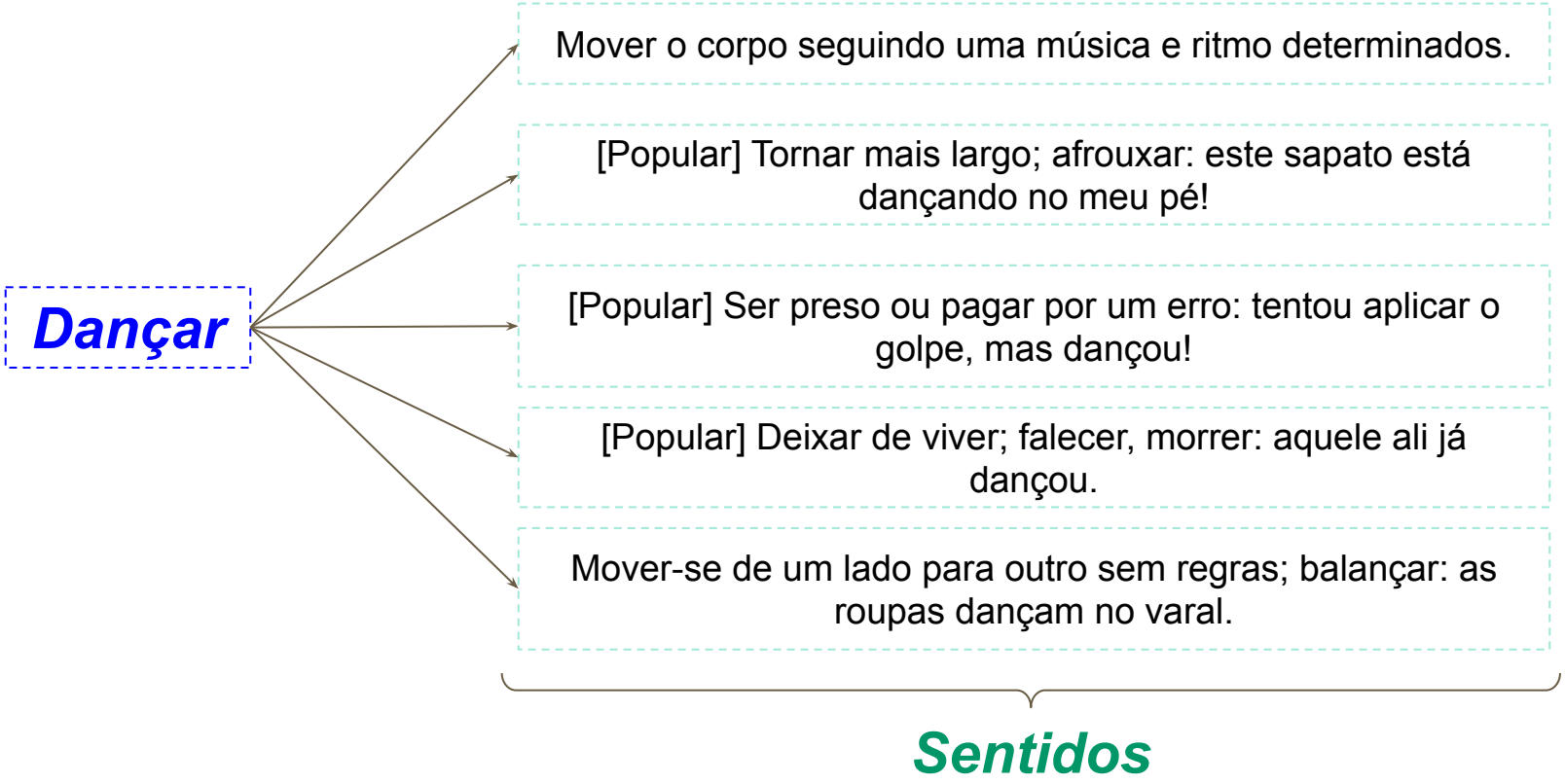




Exemplo

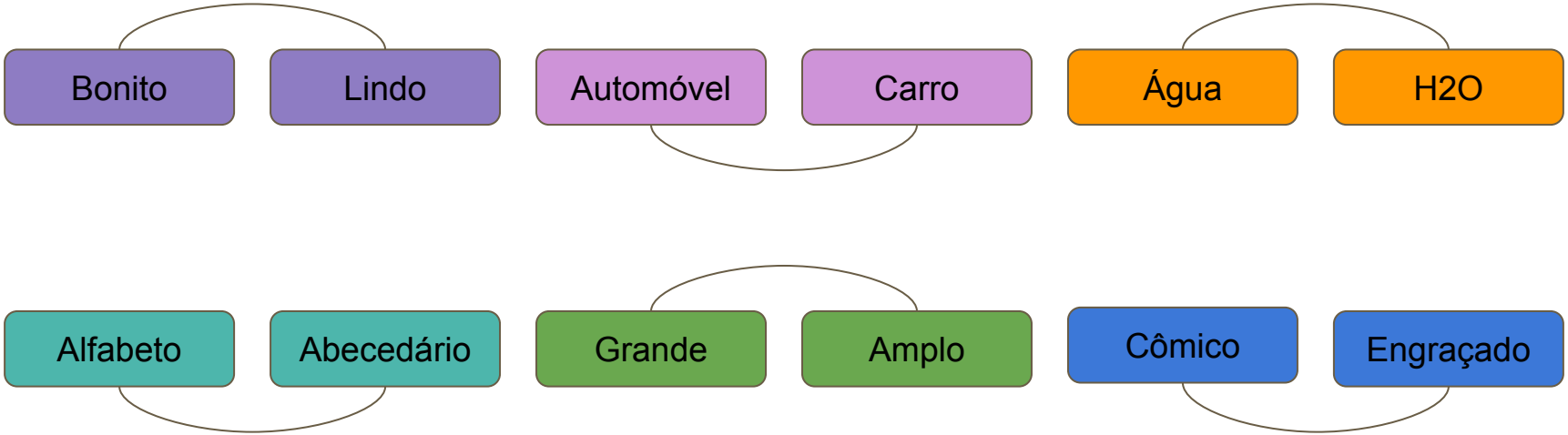

Mangueira
Árvore da família das anacardiáceas.
Tubo flexível, de tecido, borracha ou plástico, destinado a conduzir líquidos ou gases; manga, borracha.
Tradicional escola de samba brasileira da cidade do Rio de Janeiro
Bairro da cidade de Salvador-BA, situa-se na cidade baixa entre massaranduba e bonfim.
Curral grande. Local de trabalhos com gado manso e bravo, localizado próximo à casa principal e feito de pedra, pau a pique, varas etc.
Sentidos

Exemplo


Dançar
Mover o corpo seguindo uma música e ritmo determinados.
[Popular] Tornar mais largo; afrouxar: este sapato está dançando no meu pé!
[Popular] Ser preso ou pagar por um erro: tentou aplicar o golpe, mas dançou!
[Popular] Deixar de viver; falecer, morrer: aquele ali já dançou.
Mover-se de um lado para outro sem regras; balançar: as roupas dançam no varal.
Sentidos

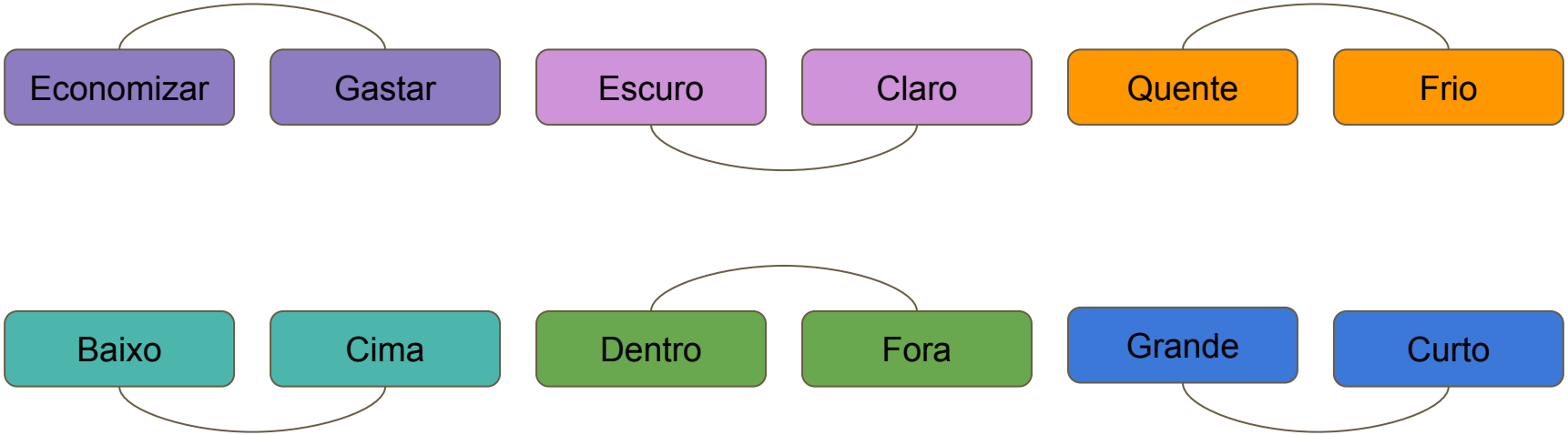
Existem relações entre os *Sentidos* das palavras.

Sinonímia



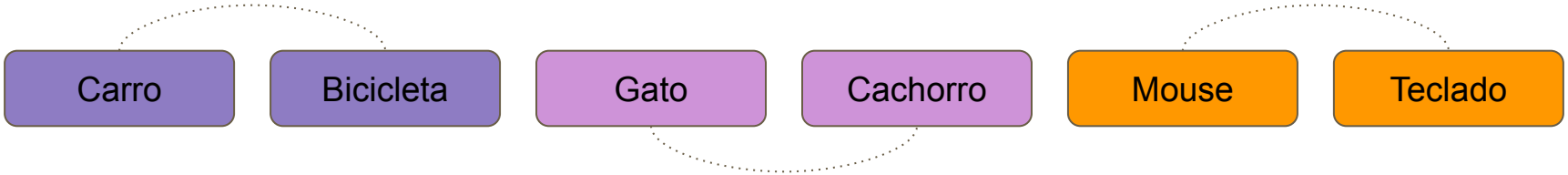
Existem relações entre os *Sentidos* das palavras.

Antonímia



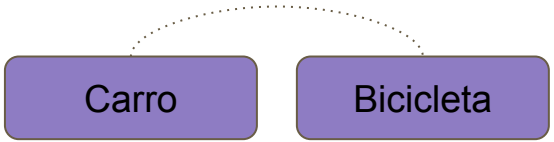
Existem relações entre os *Sentidos* das palavras.

Similaridade Não são sinônimos, mas compartilham algum elemento de significado.

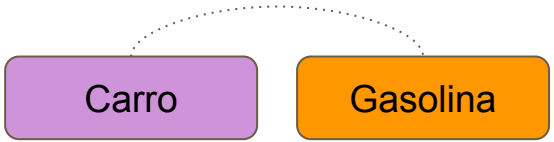


Existem relações entre os *Sentidos* das palavras.

Palavras Relacionadas



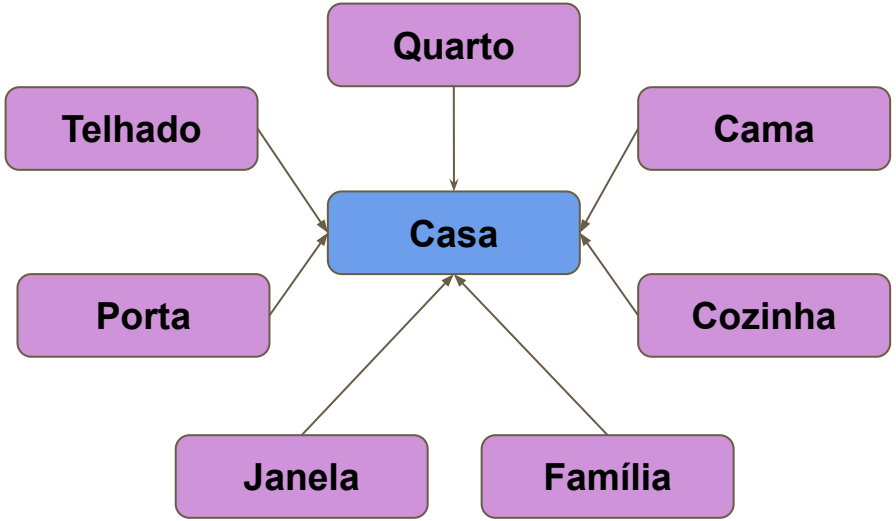
Similares



Relacionadas, mas não similares

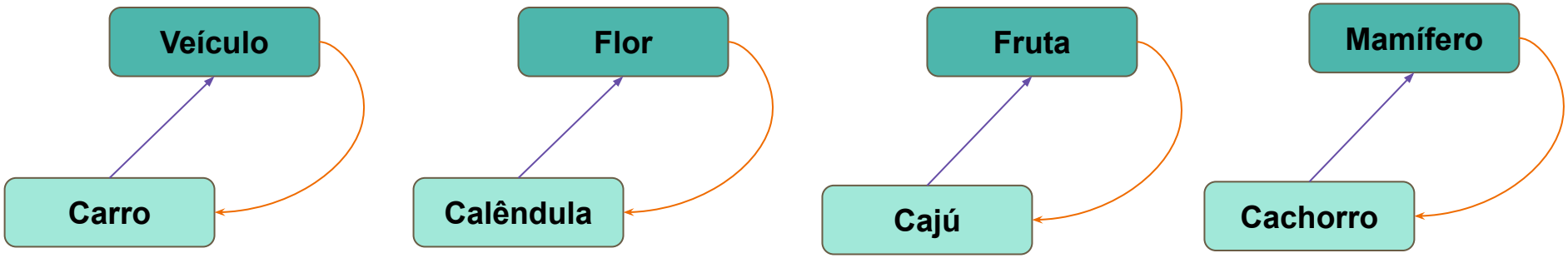
Existem relações entre os *Sentidos* das palavras.

Campos Semânticos



Existem relações entre os *Sentidos* das palavras.

Hipônimo **Hiperônimo**



Mas como definir um **conceito**?

Os **conceitos** das palavras têm associações complexas.

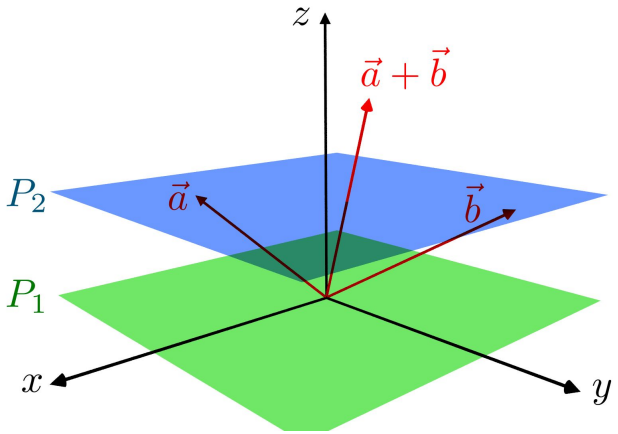
Sinonímia | **Antonímia** | **Similaridade** | **Campos Semânticos** | **Hiperônimo/ Hipônimo**

Os **conceitos** dependem do **contexto**.

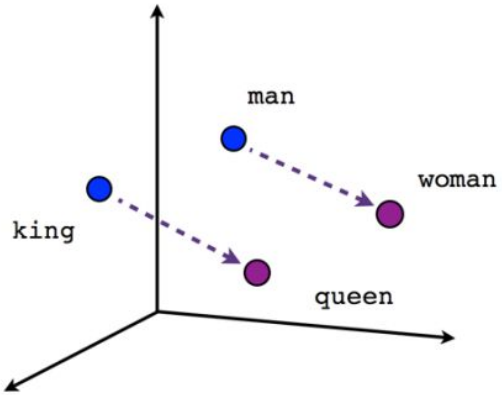
- Seria possível representar uma linguagem natural por um modelo genérico (computacional) que capturasse essas características através da análise de muitos contextos?

- Seria possível representar uma linguagem natural por um modelo genérico (computacional) que capturasse essas características através da análise de muitos contextos?

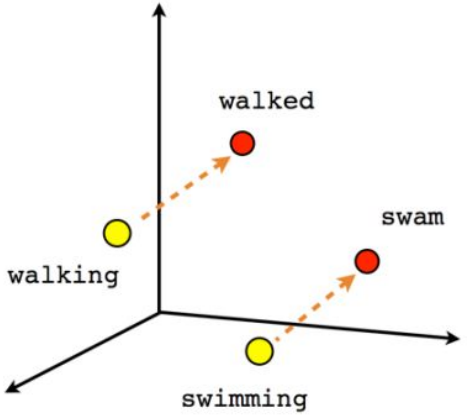
- SIM, podemos representar uma linguagem natural por um modelo de natureza matemática!



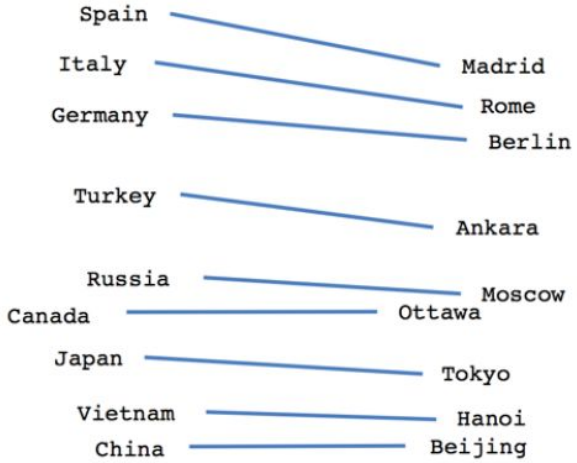
avó -0.35911 0.64988 -0.41623 0.57159 -0.38013 0.38985 -0.73346 -0.30124
espelho -0.16847 1.087 -0.34621 0.2506 -0.38312 -0.49646 -0.32659
carro -0.47797 1.1822 -0.30762 0.51227 -0.74199 0.29385 -0.19629 -0.21463
arte 0.18662 0.8713 -0.90179 0.26126 -0.19692 -0.12022 -0.34776 -0.067401



Male-Female



Verb tense



Country-Capital

Mikolov et al. 2013

Fundamentos matemáticos

Van Gogh, o ___???___

O gato sentou no ___???___

O astrônomo olha para o ___???___

Van Gogh, o ___???___

O gato sentou no ___???___

O astrônomo olha para o ___???___

PROBABILIDADE!

Van Gogh, o ___???___

O gato sentou no ___???___

O astrônomo olha para o ___???___

$$P(?|o, astrônomo, olha, para, o)$$

$$P(w_t|w_1, ..., w_n), n \in \mathbb{N}$$

$$P(céu|o, astrônomo, olha, para, o) =$$

$$= \frac{\#(\text{o astrônomo olha para o céu})}{\#(\text{o astrônomo olha para o})}$$

$$
\begin{aligned}
P(X_1 \ldots X_n) &= P(X_1)P(X_2|X_1)P(X_3|X_1^2)\ldots P(X_n|X_1^{n-1}) \\
&= \prod_{k=1}^{n} P(X_k|X_1^{k-1})
\end{aligned}
$$

$$
\begin{aligned}
P(w_1^n) &= P(w_1)P(w_2|w_1)P(w_3|w_1^2)\ldots P(w_n|w_1^{n-1}) \\
&= \prod_{k=1}^{n} P(w_k|w_1^{k-1})
\end{aligned}
$$

$$
\begin{aligned}
P(X_1 \ldots X_n) &= P(X_1)P(X_2|X_1)P(X_3|X_1^2)\ldots P(X_n|X_1^{n-1}) \\
&= \prod_{k=1}^{n} P(X_k|X_1^{k-1})
\end{aligned}
$$

$$
\begin{aligned}
P(w_1^n) &= P(w_1)P(w_2|w_1)P(w_3|w_1^2)\ldots P(w_n|w_1^{n-1}) \\
&= \prod_{k=1}^{n} P(w_k|w_1^{k-1})
\end{aligned}
$$

Mas e se w_1^n nunca acontece?

$$P(céu|o, astrônomo, olha, para, o) \approx P(céu|o)$$

$$P(w_n|w_1^{n-1}) \approx P(w_n|w_{n-1})$$

$$P(céu|o, astrônomo, olha, para, o) \approx P(céu|o)$$

$$P(w_n|w_1^{n-1}) \approx P(w_n|w_{n-1})$$

Uma sequência pode ser então estimada por:

$$\begin{aligned} P(w_1^n) &= P(w_1)P(w_2|w_1)P(w_3|w_1^2)\dots P(w_n|w_1^{n-1}) \\ &= \prod_{k=1}^{n} P(w_k|w_1^{k-1}) \\ P(w_1^n) &\approx \prod_{k=1}^{n} P(w_k|w_{k-1}) \end{aligned}$$

$$P(céu|o, astrônomo, olha, para, o) \approx P(céu|o)$$

$$P(w_n|w_1^{n-1}) \approx P(w_n|w_{n-1})$$

Uma sequência pode ser então estimada por:

$$\begin{aligned} P(w_1^n) &= P(w_1)P(w_2|w_1)P(w_3|w_1^2)\dots P(w_n|w_1^{n-1}) \\ &= \prod_{k=1}^{n} P(w_k|w_1^{k-1}) \end{aligned}$$

$$P(w_1^n) \approx \prod_{k=1}^{n} P(w_k|w_{k-1})$$

Olhando para o **Histórico**, onde estamos?

Histórico

2.500 sentenças
14 milhões de tokens

Feedforward Neural Network

1987 — 1996 — 2000 — 2003

Slava M. Katz

"Estimation of Probabilities from Sparse Data for the Language Model Component of a Speech Recognizer"

Stanley and Joshua

"An Empirical Study of Smoothing Techniques for Language Modeling"

Xu and Rudnick

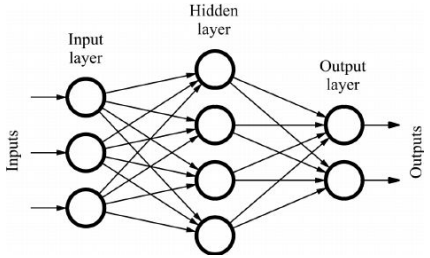
"Can Artificial Neural Networks Learn Language Models?"

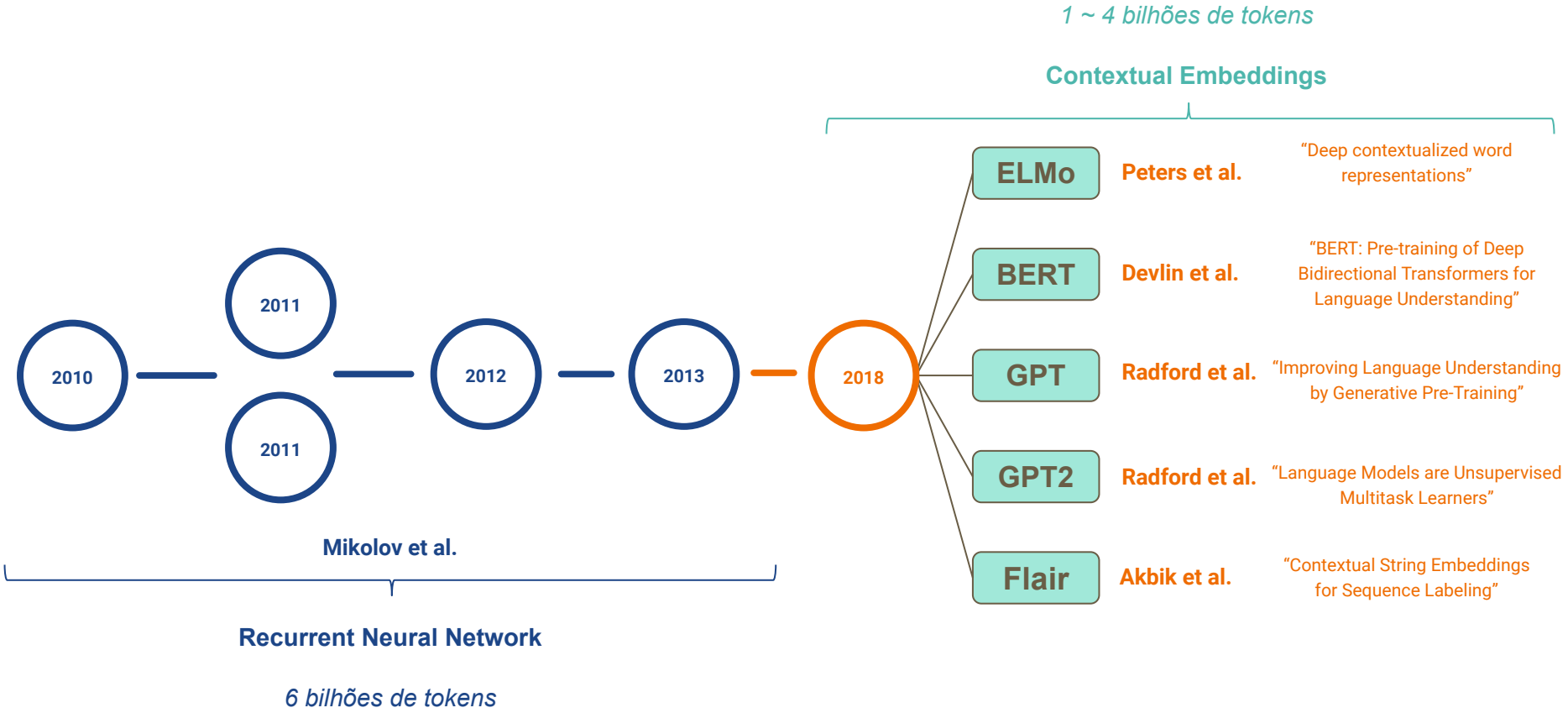
Bengio et al.

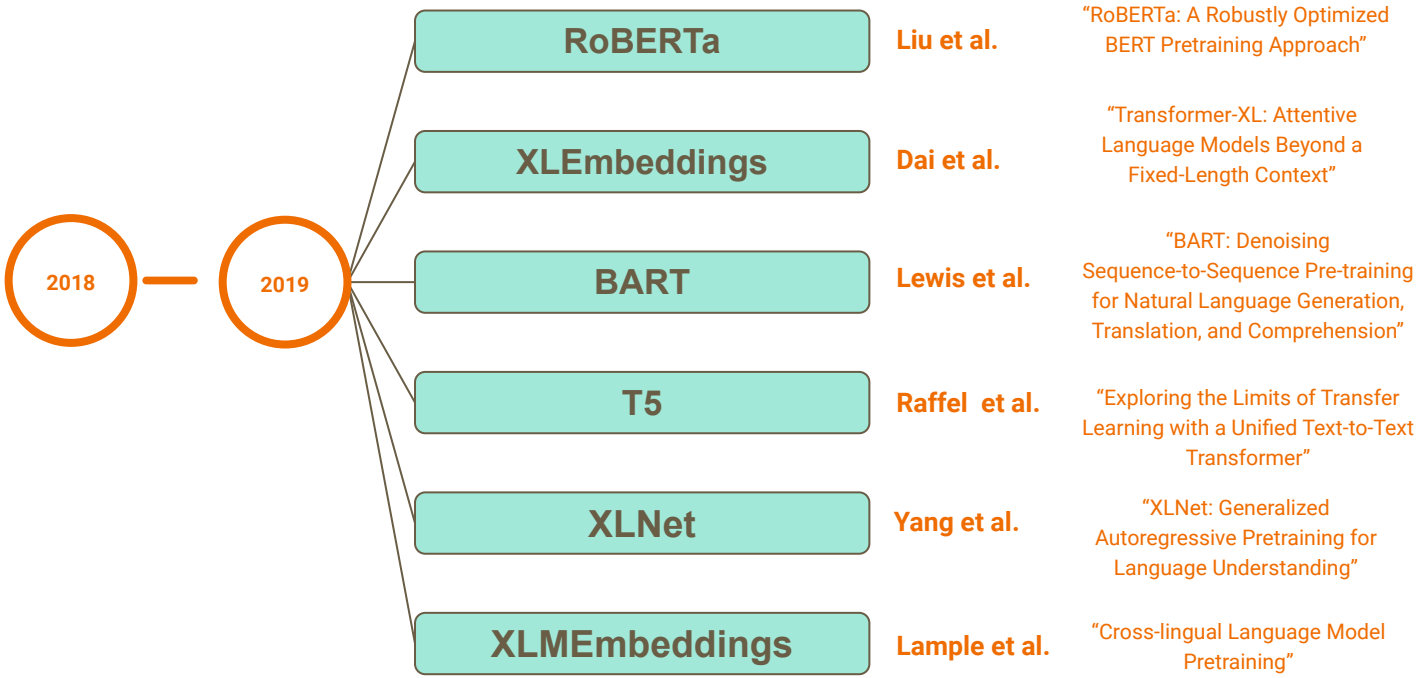
"A Neural Probabilistic Language Model"

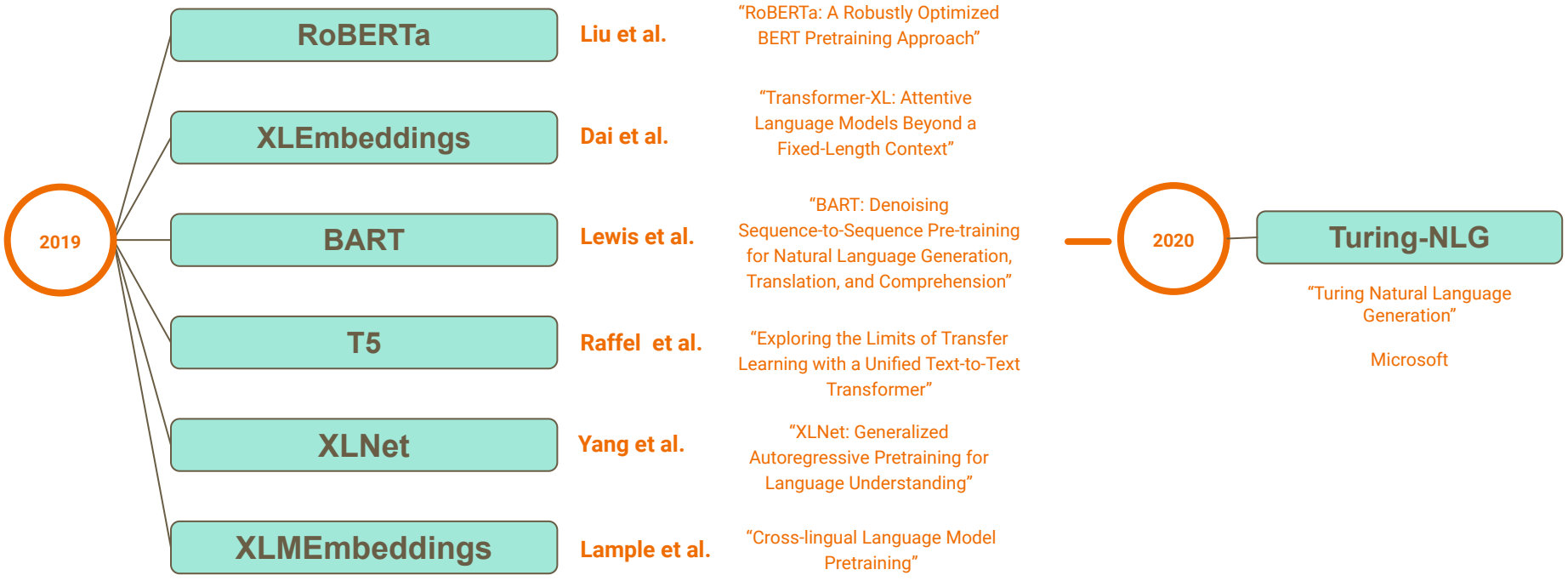
N-gram Language Models

2,3-gram
750.000 mil ~ 250 milhões tokens



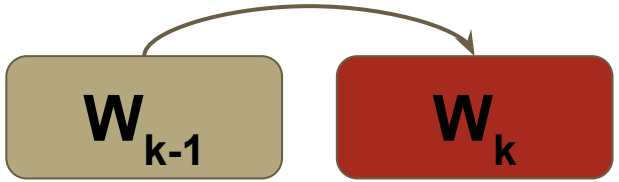

1 ~ 4 bilhões de tokens
Contextual Embeddings
ELMo
Peters et al.
"Deep contextualized word representations"
BERT
Devlin et al.
"BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding"
GPT
Radford et al.
"Improving Language Understanding by Generative Pre-Training"
GPT2
Radford et al.
"Language Models are Unsupervised Multitask Learners"
Flair
Akbik et al.
"Contextual String Embeddings for Sequence Labeling"
2010
2011
2011
2012
2013
2018
Mikolov et al.
Recurrent Neural Network
6 bilhões de tokens


2018
2019
RoBERTa
Liu et al.
"RoBERTa: A Robustly Optimized BERT Pretraining Approach"
XLEmbeddings
Dai et al.
"Transformer-XL: Attentive Language Models Beyond a Fixed-Length Context"
BART
Lewis et al.
"BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension"
T5
Raffel et al.
"Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer"
XLNet
Yang et al.
"XLNet: Generalized Autoregressive Pretraining for Language Understanding"
XLMEmbeddings
Lample et al.
"Cross-lingual Language Model Pretraining"


2019
RoBERTa
Liu et al.
"RoBERTa: A Robustly Optimized BERT Pretraining Approach"
XLEmbeddings
Dai et al.
"Transformer-XL: Attentive Language Models Beyond a Fixed-Length Context"
BART
Lewis et al.
"BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension"
T5
Raffel et al.
"Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer"
XLNet
Yang et al.
"XLNet: Generalized Autoregressive Pretraining for Language Understanding"
XLMEmbeddings
Lample et al.
"Cross-lingual Language Model Pretraining"
2020
Turing-NLG
"Turing Natural Language Generation"
Microsoft

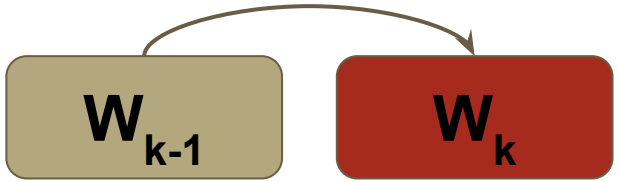
Modelos WE iniciais

Nós vimos que uma sequência pode ser então estimada por: $P(w_1^n) \approx \prod_{k=1}^{n} P(w_k|w_{k-1})$

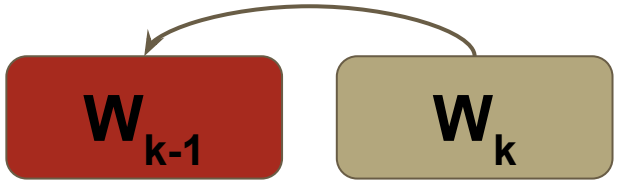


$$P(w_k|w_{k-1})$$

Nós vimos que uma sequência pode ser então estimada por: $P(w_1^n) \approx \prod_{k=1}^{n} P(w_k|w_{k-1})$



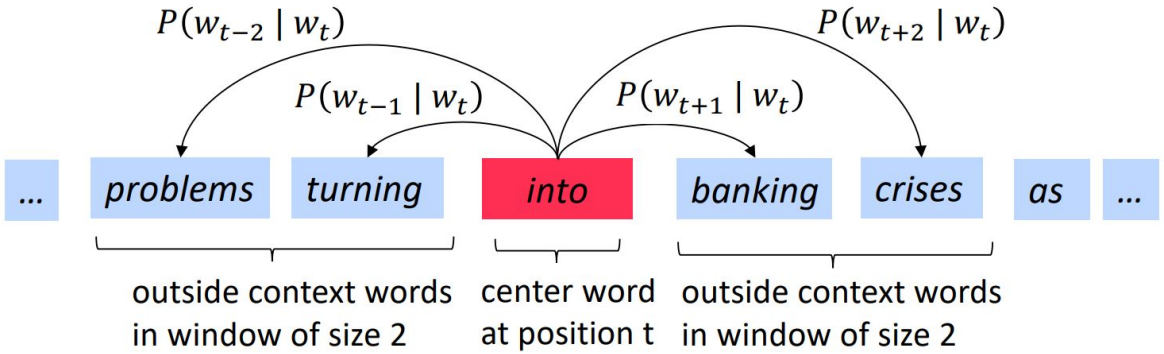
$$P(w_k|w_{k-1})$$



$$P(w_{k-1}|w_k)$$

Nós vimos que uma sequência pode ser então estimada por: $P(w_1^n) \approx \prod_{k=1}^{n} P(w_k|w_{k-1})$

$$P(w_t|w_1, ..., w_n), n \in \mathbb{N} \quad \textbf{e} \quad P(w_1, ..., w_n|w_t), n \in \mathbb{N}$$



De outro modo: Para cada posição t, prevemos palavras de contexto.

$$L(\theta) = \prod_{t=1}^{T} \prod_{-m \leq j \leq m} P(w_{t+j} | w_t; \theta)$$

Temos que maximizar essa probabilidade.

Negative Log Likelihood (NLL) $J(\theta) = -\frac{1}{T} log L(\theta)$

$$L(\theta) = \prod_{t=1}^{T} \prod_{-m \leq j \leq m} P(w_{t+j}|w_t;\theta) \qquad J(\theta) = -\frac{1}{t} \sum_{t=1}^{T} \sum_{-m \leq j \leq m} logP(w_{t+J}|w_t;\theta)$$

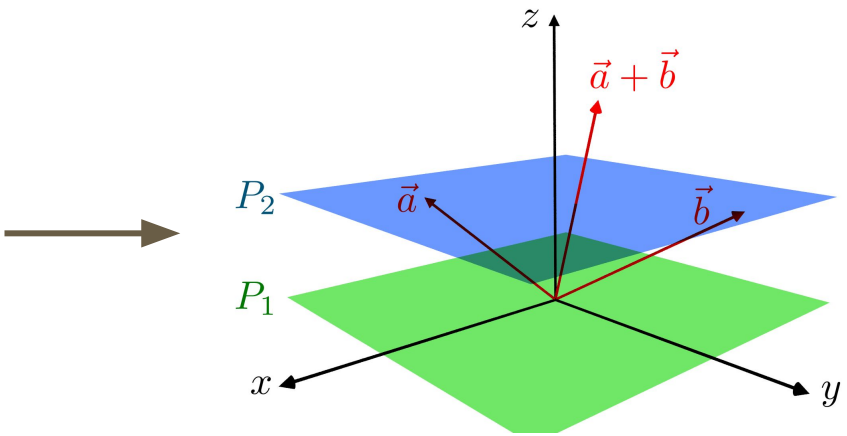
$\therefore$

$$\frac{\partial}{\partial v_c} J(\theta)$$

Minimizar a função objetivo

⇔

Maximizar a precisão preditiva

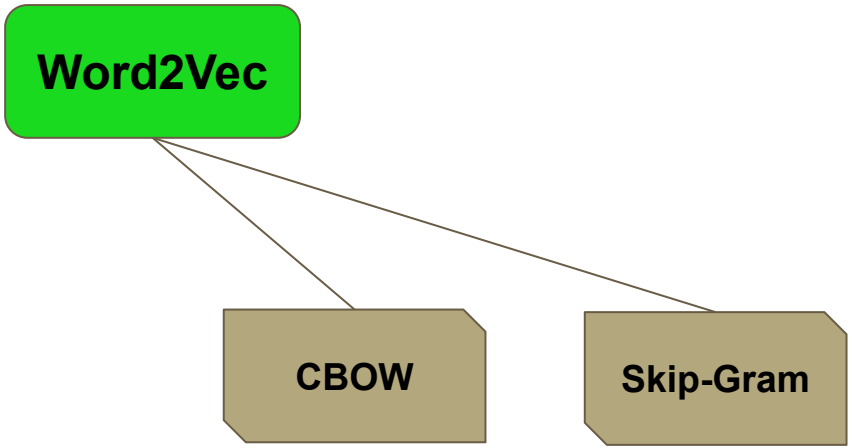


avó -0.35911 0.64988 -0.41623 0.57159 -0.38013 0.38985 -0.73346 -0.30124
espelho -0.16847 1.087 -0.34621 0.2506 -0.38312 -0.49646 -0.32659
carro -0.47797 1.1822 -0.30762 0.51227 -0.74199 0.29385 -0.19629 -0.21463
arte 0.18662 0.8713 -0.90179 0.26126 -0.19692 -0.12022 -0.34776 -0.067401

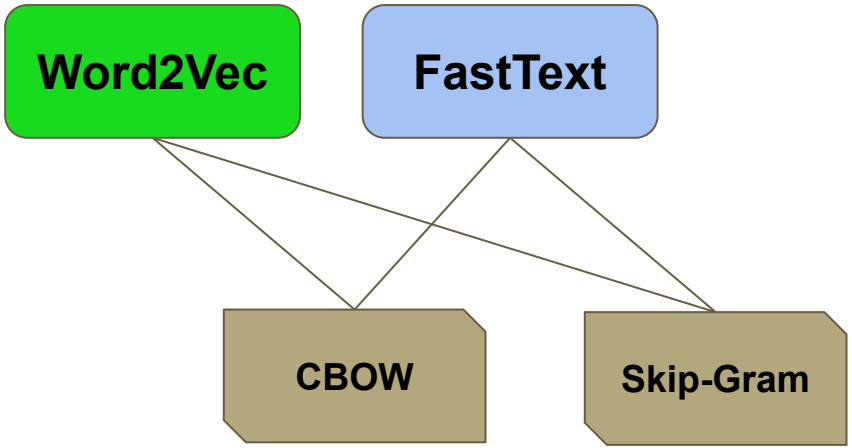
- Existem várias arquiteturas de Rede Neural que geram Word Embeddings:

Word2Vec

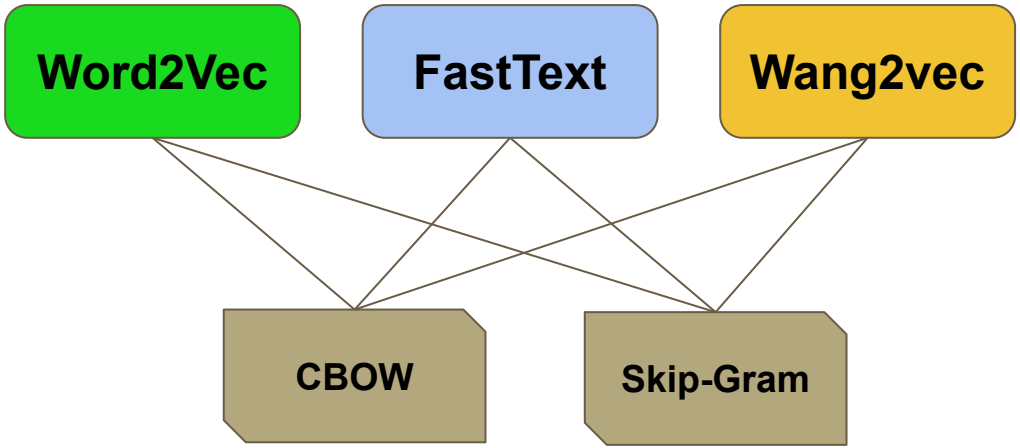
- Existem várias arquiteturas de Rede Neural que geram Embeddings:



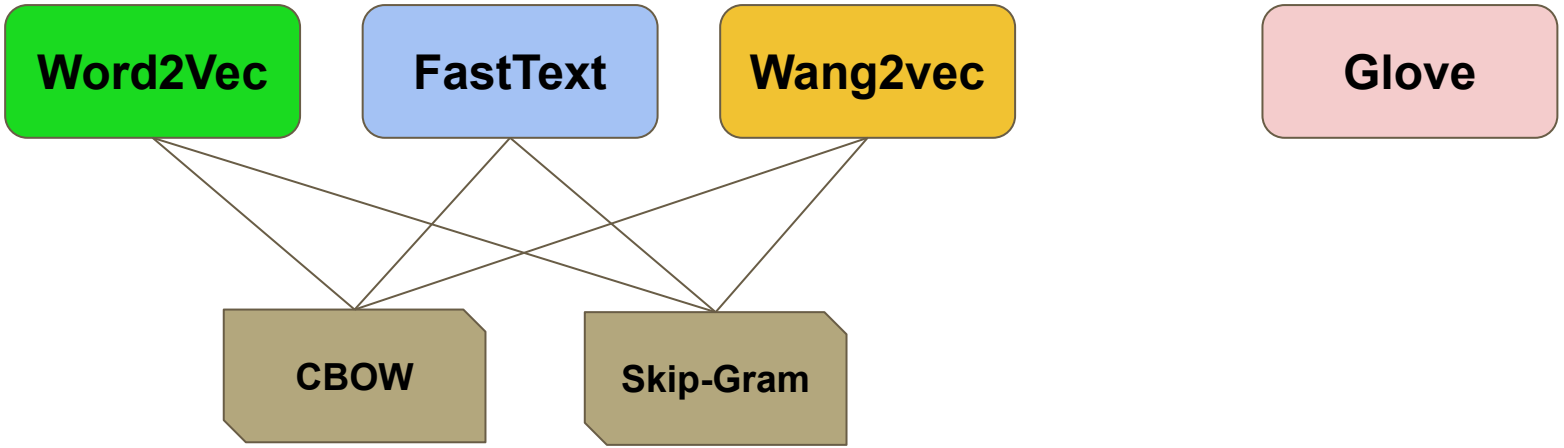
- Existem várias arquiteturas de Rede Neural que geram Embeddings:



- Existem várias arquiteturas de Rede Neural que geram Embeddings:

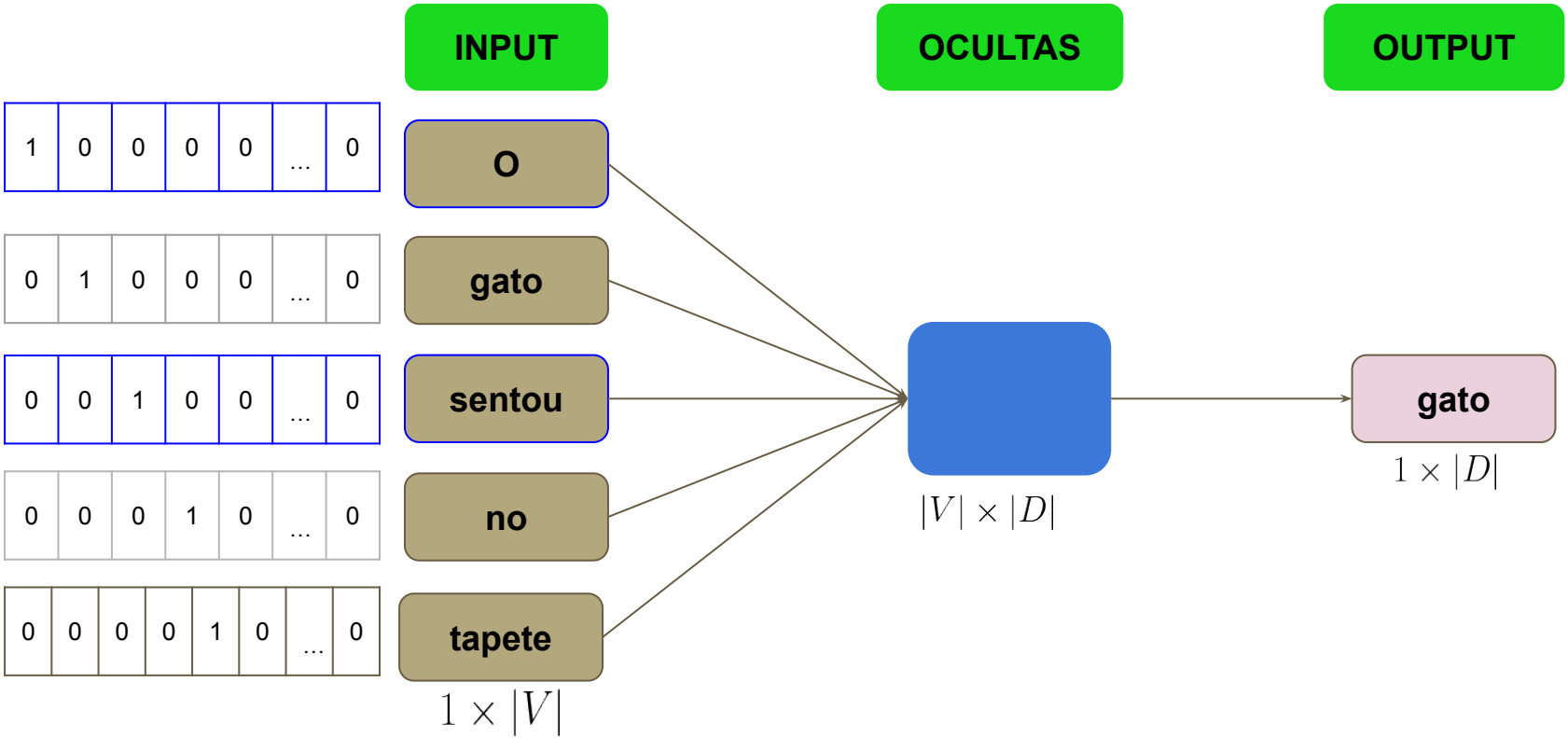


- Existem várias arquiteturas de Rede Neural que geram Embeddings:



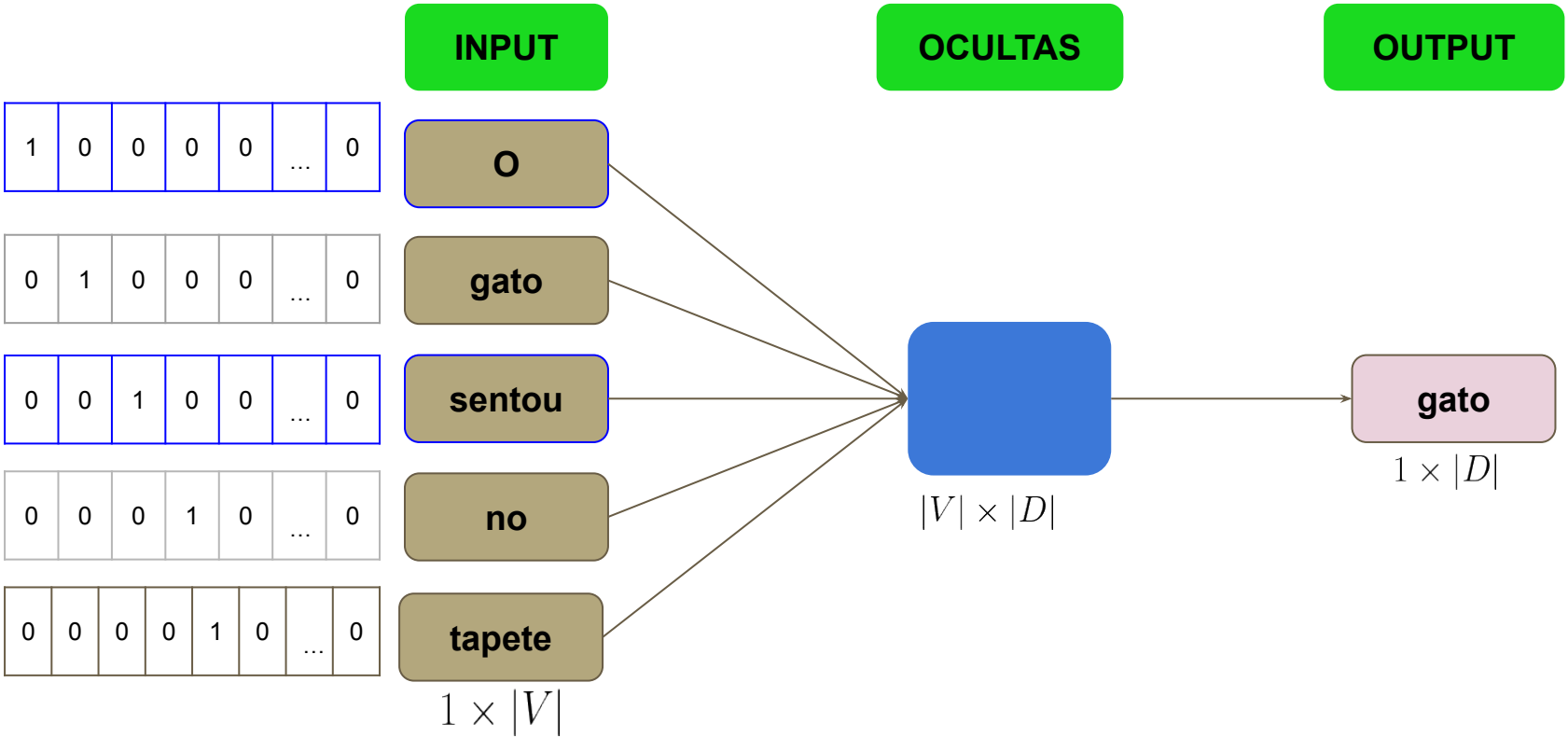
CBOW - Continuous Bag-Of-Words

O gato sentou no tapete



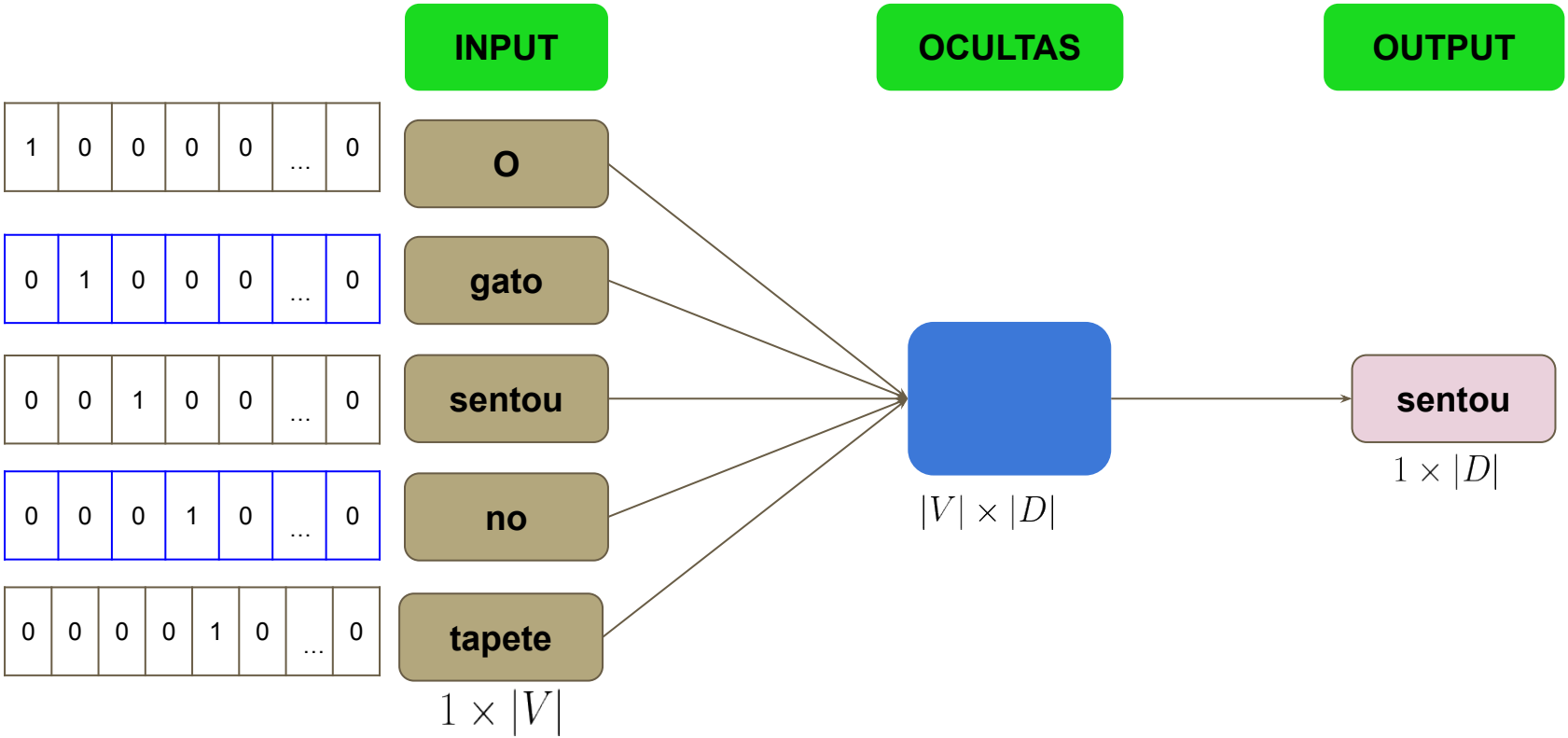
CBOW - Continuous Bag-Of-Words

O gato sentou no tapete



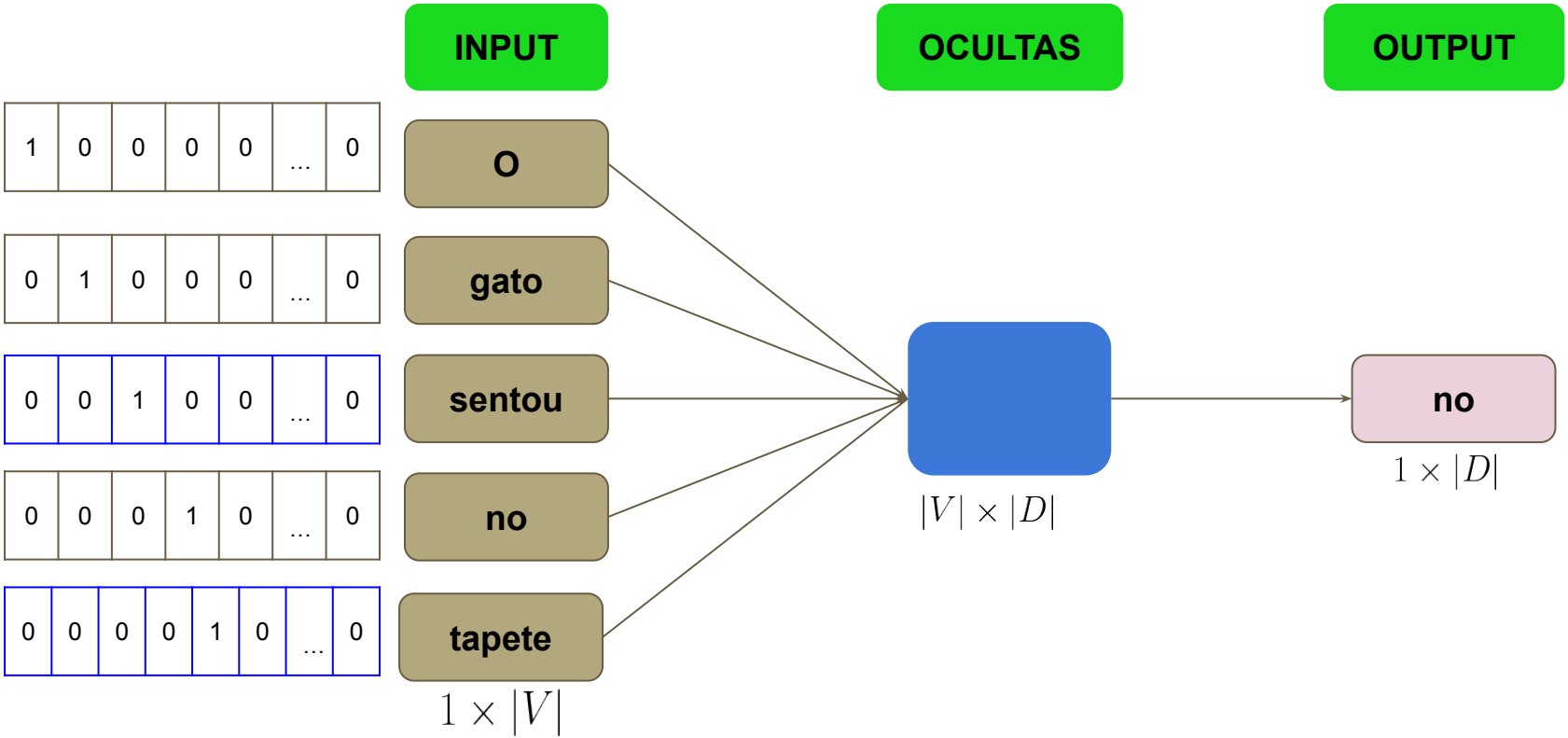
CBOW - Continuous Bag-Of-Words

O gato sentou no tapete



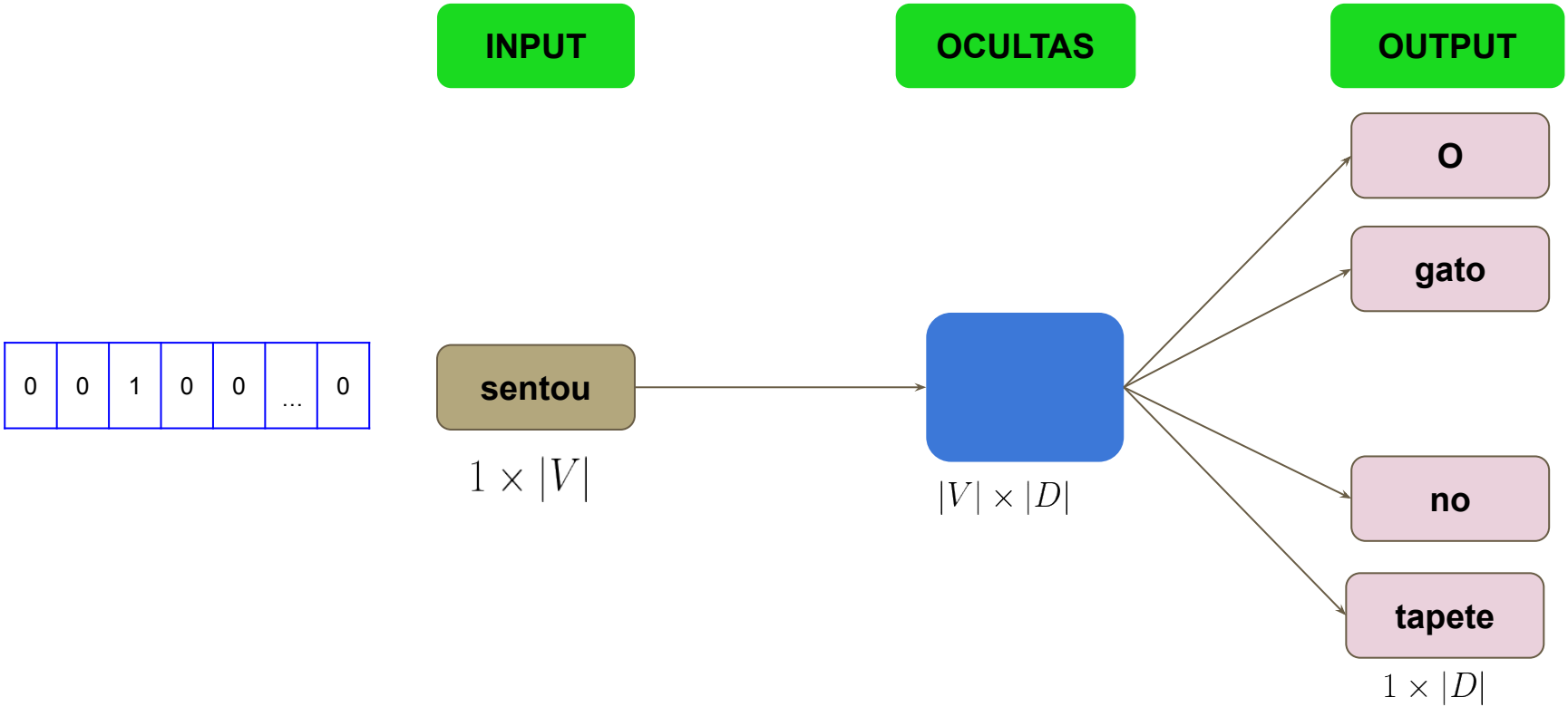
CBOW - Continuous Bag-Of-Words

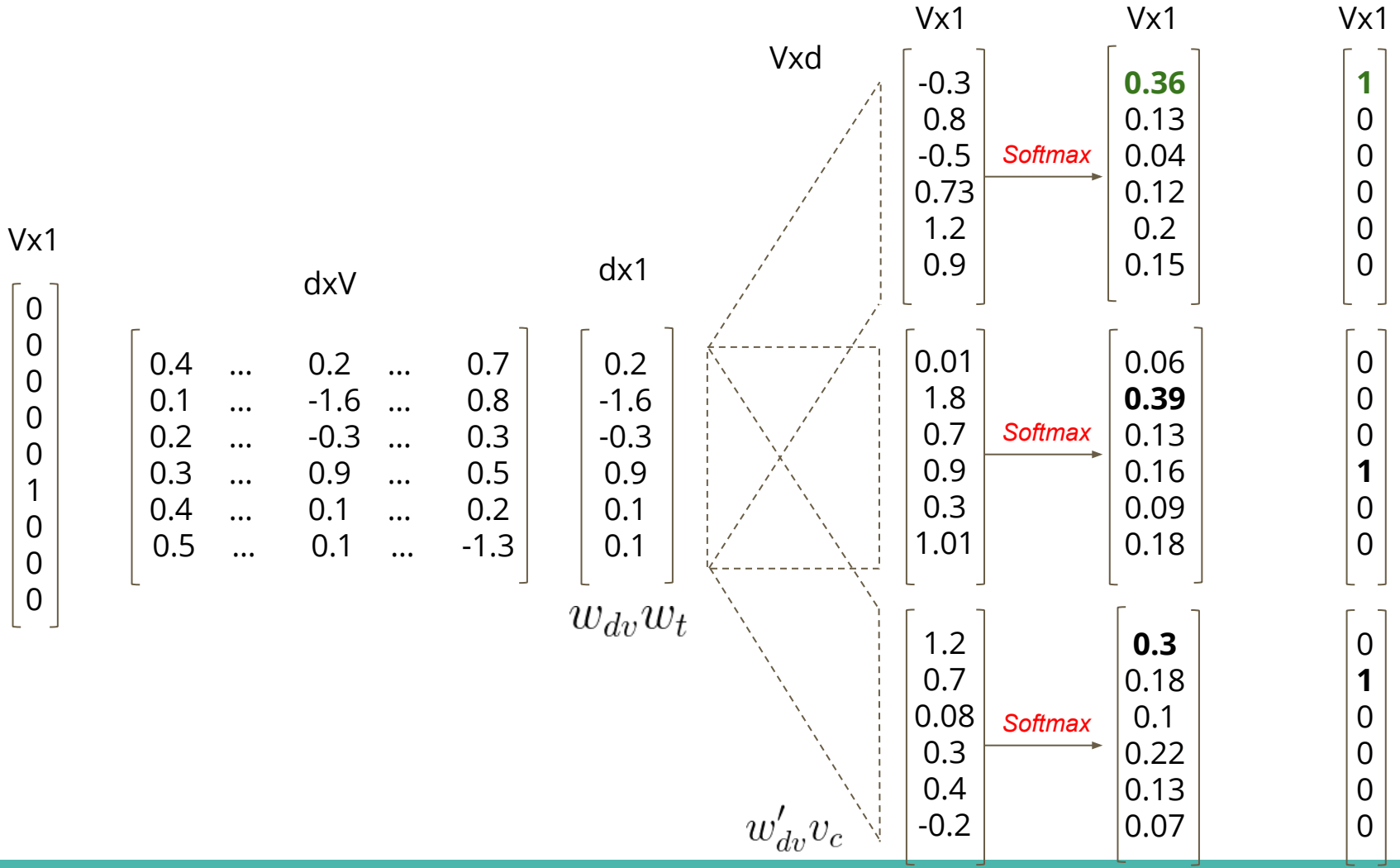
O gato sentou no tapete



Skip-Gram

O gato sentou no tapete




Vx1
0
0
0
0
1
0
0
0
dxV
0.4 ... 0.2 ... 0.7
0.1 ... -1.6 ... 0.8
0.2 ... -0.3 ... 0.3
0.3 ... 0.9 ... 0.5
0.4 ... 0.1 ... 0.2
0.5 ... 0.1 ... -1.3
dx1
0.2
-1.6
-0.3
0.9
0.1
0.1
$w_{dv}w_t$
Vxd
$w'_{dv}v_c$
Vx1
-0.3
0.8
-0.5
0.73
1.2
0.9
Softmax
Vx1
0.36
0.13
0.04
0.12
0.2
0.15
Vx1
1
0
0
0
0
0
0.01
1.8
0.7
0.9
0.3
1.01
Softmax
0.06
0.39
0.13
0.16
0.09
0.18
0
0
0
1
0
0
1.2
0.7
0.08
0.3
0.4
-0.2
Softmax
0.3
0.18
0.1
0.22
0.13
0.07
0
1
0
0
0
0

Processo de construção do modelo

- Construir o vocabulário do corpus
- Gerar o input para a arquitetura (context, target)
- Construir a arquitetura do modelo
- Treinar o Modelo
- Obter os embeddings

CBOW e Skip-Gram

avó -0.35911 0.64988 -0.41623 0.57159 -0.38013 0.38985 -0.73346 -0.30124
espelho -0.16847 1.087 -0.34621 0.2506 -0.38312 -0.49646 -0.32659
carro -0.47797 1.1822 -0.30762 0.51227 -0.74199 0.29385 -0.19629 -0.21463
arte 0.18662 0.8713 -0.90179 0.26126 -0.19692 -0.12022 -0.34776 -0.067401

Word2Vec

- Desenvolvido por pesquisadores da Google;
- Possibilidade de dois tipos de treinamento:
  - Skip-Gram
  - CBOW

FastText

- Desenvolvido por pesquisadores do Facebook;
- Consegue aproximar um valor para palavras não contidas no vocabulário;
- Cada palavra é representado por uma coleção de n-gramas:

renoir = <re, ren, eno, noi, oir, ir>

- FastText tem sido usado para diversas tarefas, entre elas classificação de texto e REN [Bojanowski, 2016];

Wang2Vec

- Desenvolvido por pesquisadores do INESC-ID (Portugal) e CMU (EUA);
- Modificação no Word2Vec (Skip-Gram e CBOW) para tarefas envolvendo sintaxe;
- Impulsionaram resultados para Part-Of-Speech Tagging e Dependency Parsing [Ling, 2015];

Glove

- Desenvolvido por pesquisadores de Stanford;
- Utiliza uma Matriz de co-ocorrência entre palavras em um contexto para aprender seu significado;
- As probabilidades de predição são calculadas com base na matriz de co-ocorrência [Pennington, 2014];

Modelos dinâmicos, sensíveis ao contexto

Motivação

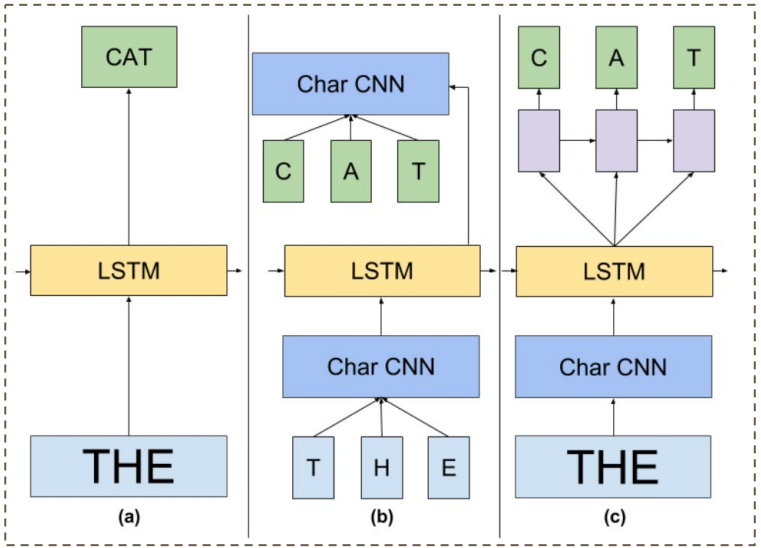
Mangueira *tensor([-0.0365, -0.1508, 0.0065, ..., 0.1447, 0.1526, -0.1346])*

A --> tensor([0.0234, -0.0244, -0.0039, ..., 0.1887, 0.0441, -0.0241])
mangueira --> tensor([0.0610, -0.1249, -0.0066, ..., 0.1447, 0.1526, -0.1346])
está --> tensor([0.0017, -0.0064, 0.0015, ..., -0.1571, -0.2378, -0.0496])
cheia --> tensor([0.1113, -0.0066, 0.0196, ..., -0.1280, 0.1558, -0.2580])
de --> tensor([0.2375, 0.0074, -0.0143, ..., 0.0438, 0.0729, -0.1007])
frutos! --> tensor([-0.0756, -0.0022, -0.0008, ..., 0.0000, 0.0000, 0.0000])

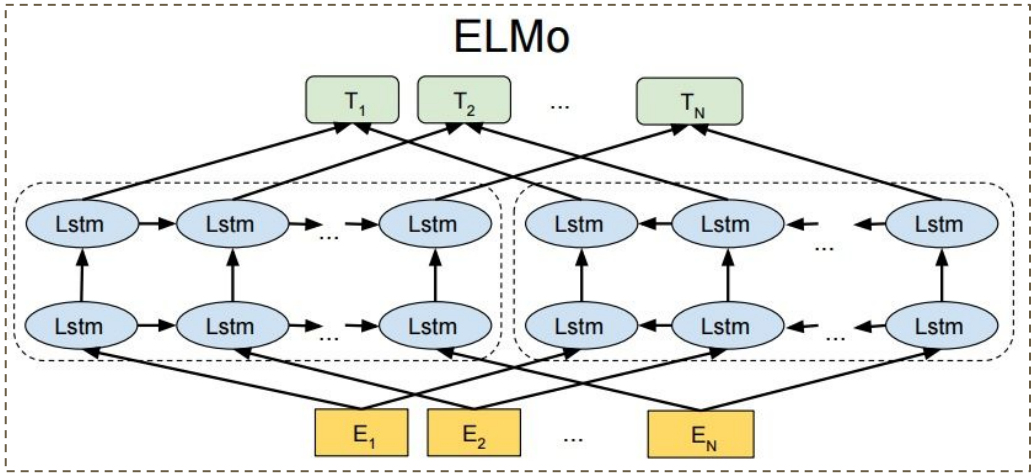
Quanto --> tensor([0.0316, 0.0026, 0.0140, ..., -0.3645, 0.1062, -0.1547])
custa --> tensor([2.4523e-01, -1.5203e-05, 9.3146e-03, ..., 9.8593e-02, 1.5959e-01, 2.2675e-01])
a --> tensor([-0.1728, -0.0290, 0.0043, ..., 0.1887, 0.0441, -0.0241])
mangueira --> tensor([-0.0845, -0.1070, -0.0008, ..., 0.1447, 0.1526, -0.1346])
do --> tensor([0.0919, 0.0257, -0.0442, ..., 0.1748, -0.0800, -0.1993])
radiador? --> tensor([2.0074e-01, -1.4746e-02, 1.7419e-04, ..., 0.0000e+00, 0.0000e+00, 0.0000e+00])

Hoje --> tensor([0.0342, -0.0047, 0.0275, ..., 0.0495, -0.0992, -0.0210])
será --> tensor([-0.0833, -0.0167, 0.0071, ..., -0.2242, -0.0804, -0.1034])
o --> tensor([-0.0836, -0.0149, 0.0045, ..., 0.1768, 0.1693, -0.1223])
desfile --> tensor([-0.0160, -0.0048, 0.0005, ..., -0.0340, 0.2525, -0.3314])
da --> tensor([-0.0571, 0.0065, -0.0605, ..., 0.1263, -0.1481, -0.1220])
Mangueira --> tensor([0.0593, -0.0889, 0.0104, ..., 0.1447, 0.1526, -0.1346])

ELMo - Embeddings from Language Models



Jozefowicz et al. 2016



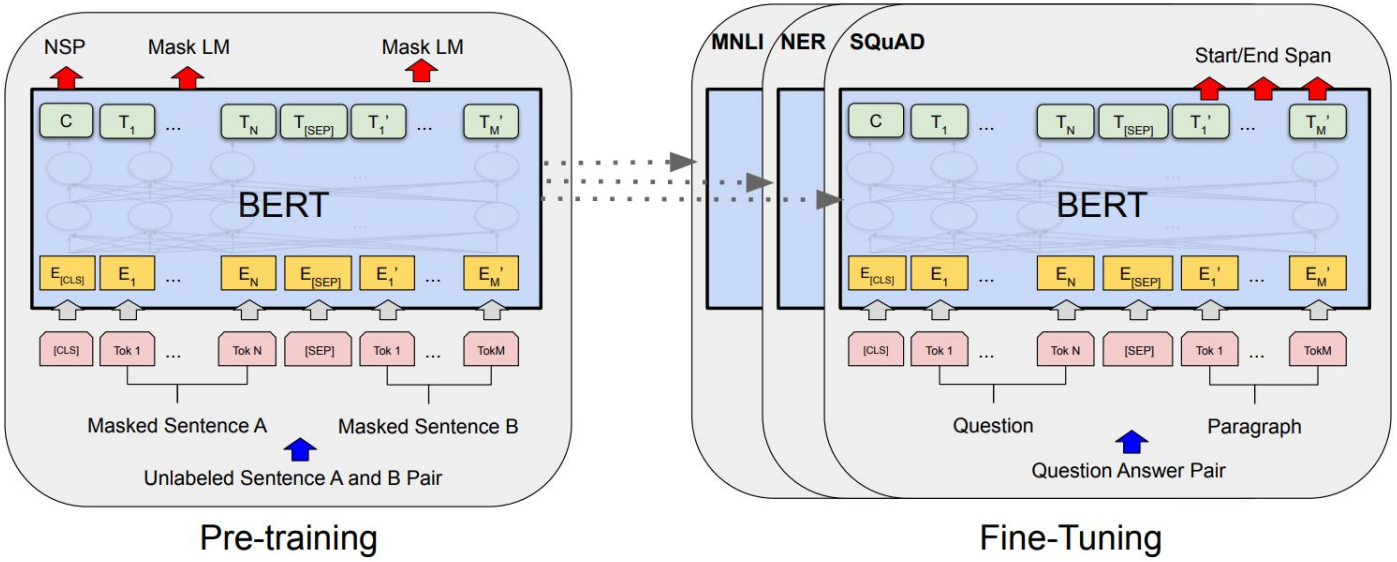
2 BiLSTM Layers com 4096 unidades
1B Tokens para Treinamento

Peters et al. 2018

BERT - **B**idirectional **E**ncoder **R**epresentations from **T**ransformers [Devlin, 2018]

- Recente modelo de representação de linguagem desenvolvido pelo Google;
- No artigo original de apresentação do BERT são apresentadas 11 tarefas de PLN onde o BERT foi aplicado; entre elas:
  - **Multi-Genre Natural Language Inference:** dado um par de sentenças deve-se decidir se a segunda sentença é uma continuação, contradição ou neutra em relação a primeira sentença;
  - **Semantic Textual Similarity Benchmark:** classifica a similaridade entre duas sentenças;
  - **Named Entity Recognition:** reconhecer em um texto determinadas menções e classificá-las em categorias;

BERT


NSP
Mask LM
Mask LM
BERT
[CLS]
Tok 1
Tok N
[SEP]
Tok 1
TokM
Masked Sentence A
Masked Sentence B
Unlabeled Sentence A and B Pair
Pre-training
MNLI
NER
SQuAD
Start/End Span
BERT
Question
Paragraph
Question Answer Pair
Fine-Tuning

BERT

- Treinar um modelo BERT envolve duas tarefas:

Masked LM

Next Sentence Prediction

Masked LM: o objetivo é "mascarar" 15% dos tokens de uma sentença para predição. Algumas regras:

- 80% das vezes: a palavra é substituída pelo símbolo [MASK], por exemplo:

  `meu cachorro é cabeludo. → meu [MASK] é cabeludo.`

- 10% das vezes: a palavra é substituída por uma palavra aleatória, por exemplo:

  `meu cachorro é cabeludo. → meu cachorro é maçã.`

- 10% das vezes: a palavra é mantida, por exemplo:

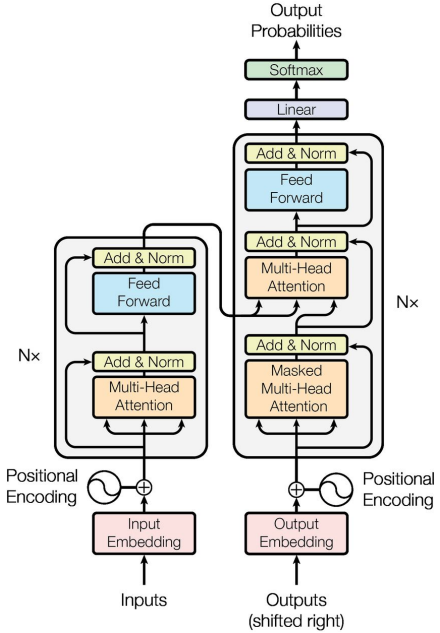
  `meu cachorro é cabeludo. → meu cachorro é cabeludo.`

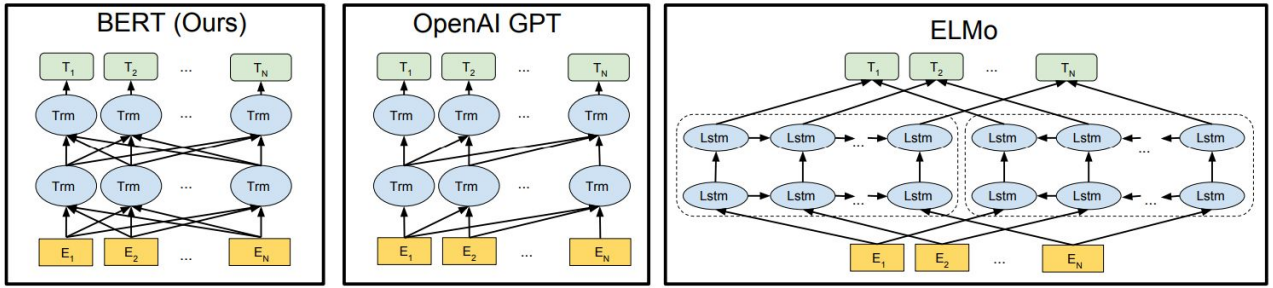
Next Sentence Prediction: dado um par de sentenças (s_1, s_2) o modelo deve predizer se a sentença s_2 é a subsequente a s_1. Algumas regras:

- 50% dos pares de sentenças são tal que s_2 é de fato a sentença subsequente;
- 50% dos outros pares são tal que s_2 é uma sentença aleatória do corpus;

BERT

A Rede Neural responsável por realizar as tarefas **Masked LM** e **Next Sentence Prediction** é composta por várias camadas do neurônio **Transformer** [Vaswani, 2017].





BERT

- Treinar um modelo de linguagem BERT requer muito recurso computacional;
- Por exemplo, para o Inglês foram usadas 16 Cloud TPUs com um corpora de 3,3Bi tokens;
- Existe um modelo Multi-lingue que contempla o **Português**;

Tipo	Idioma	Camadas	Camadas Ocultas	Self-Attetion heads
$BERT_{BASE}$	Inglês	12	768	12
$BERT_{LARGE}$	Inglês	24	1024	16
$BERT_{BASE}$	Multi-lingue	12	768	12

Flair

Flair Embedding - é um recente modelo de embedding que permite a modelagem da linguagem com base na distribuição das sequências de caracteres e palavras [Akbik, 2018];

- A geração de um Flair Embedding **não só depende** do contexto das **palavras** vizinhas, mas também do nível de **caractere** das palavras vizinhas;
- O modelo é gerado a partir de duas redes **LSTM** que capturam e incorporam as informações aprendidas durante o treinamento;

Flair

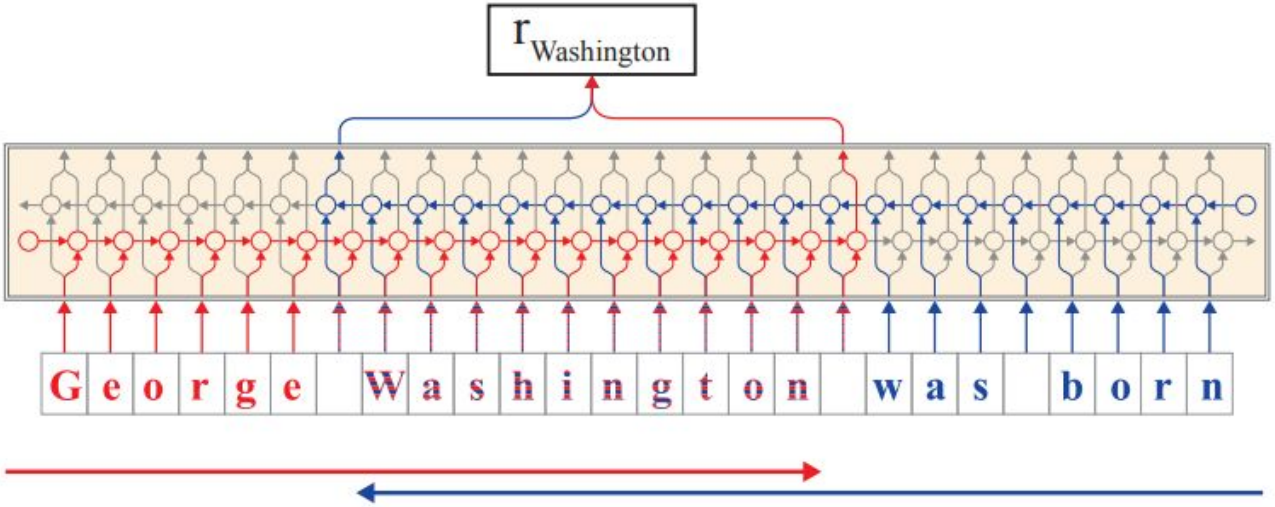
LSTM - Long Short-Term Memory - é um tipo de Rede Neural Recorrente, com uma estrutura computacional mais complexa, que tem tido sucesso na resolução de tarefas sequenciais [Tai, 2015];

- Uma LSTM permite manter, alterar ou descartar informações anteriores para relacionar com uma informação atual;

Há uma variação das redes LSTM, que são as **Bidirectional LSTM (Bi-LSTM)**;

- As redes Bi-LSTM consistem de duas LSTM que funcionam em paralelo;

Flair



Flair

- Dado $X_{0:T}$ uma sequência de caracteres, que produz uma linguagem natural;

$$X_{0:T} := (x_0, x_1, ..., x_{t-1}, x_t)$$

- Será possível prever $\{x_t\} \in X_{0:T} | X_{0:t-1} \subset X_{0:T}$?

Flair

- Dado $X_{0:T}$ uma sequência de caracteres, que produz uma linguagem natural;

$$X_{0:T} := (x_0, x_1, ..., x_{t-1}, x_t)$$

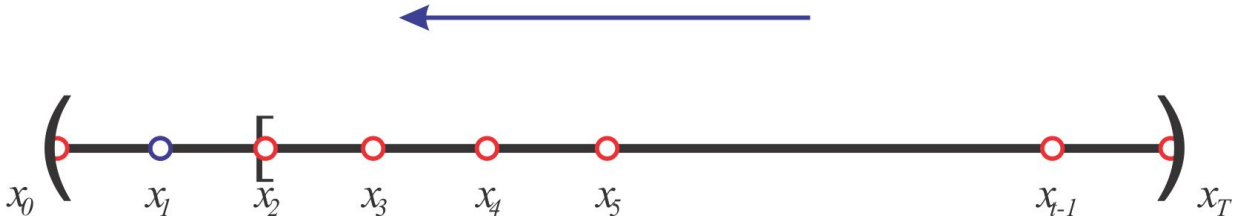
- Será possível prever $\{x_t\} \in X_{0:T} | X_{0:t-1} \subset X_{0:T}$?
- **Sim**, podemos aprender como os caracteres dessa linguagem estão distribuídos, abstraindo um modelo de linguagem [Rosenfeld, 2000];
- Então quando dado $X_{0:x_{t-1}}$ queremos predizer $\{x_t\} \in X_{0:T}$:

$$P(x_t|x_0, x_1, ..., x_{t-1}) = P(x_{0:T}) \approx \prod_{t=0}^{T} P(x_t|h_t; \theta)$$

Flair

- Então quando dado $X_{0:x_{t-1}}$ queremos predizer $\{x_t\} \in X_{0:T}$:

$$P(x_t|x_0, x_1, ..., x_{t-1}) = P(x_{0:T}) \approx \prod_{t=0}^{T} P(x_t|h_t; \theta)$$



Flair

- Como Flair Embedding é treinado por duas LSTM temos dois modelos:

$$p^f(x_t|X_{t+1:T}) \approx \prod_{t=0}^{T} p^f(x_t|h_t^f;\theta)$$
$$h_t^f = f_h^f(x_{t-1}, h_{t-1}^f, c_{t-1}^f;\theta)$$
$$c_t^f = f_c^f(x_{t-1}, h_{t-1}^f, c_{t-1}^f;\theta)$$

Forward

$$p^b(x_t|X_{t+1:T}) \approx \prod_{t=0}^{T} p^b(x_t|h_t^b;\theta)$$
$$h_t^b = f_h^b(x_{t-1}, h_{t-1}^b, c_{t-1}^b;\theta)$$
$$c_t^b = f_c^b(x_{t-1}, h_{t-1}^b, c_{t-1}^b;\theta)$$

Backward

Flair

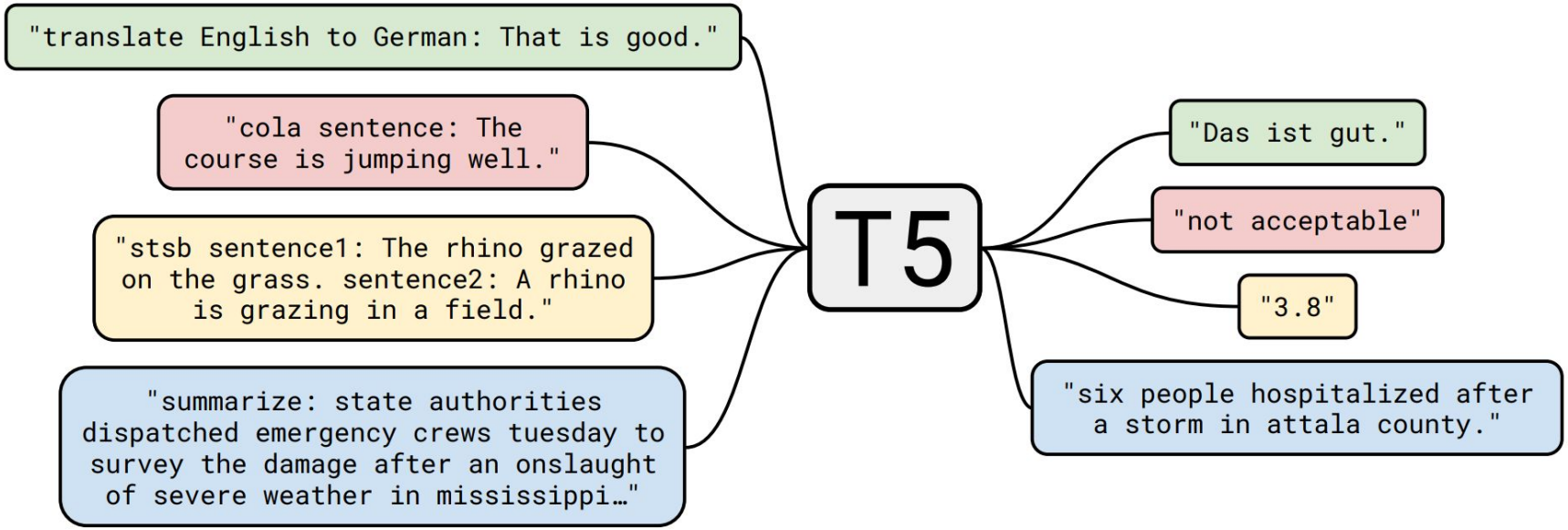
- Os processos de **Forward** e **Backward** produzem duas saídas para cada palavra: h_t^f e h_t^b ;
- Essas saídas são empilhadas em uma matriz, formando o embedding final:

$$w^{CharLM} := \begin{bmatrix} h^f_{t_{i+1}-1} \\ h^b_{t_i-1} \end{bmatrix}$$

T5

- Treinamento Supervisionado e não-supervisionado (unsupervised);
  - NLC e CV (ImageNet)
- Um dos pontos mais fortes do aprendizado não-supervisionado é não precisar de dado anotado;
- Algoritmos não-supervisionados para geração de Modelos de Linguagem;
- Um grande corpus: C4 (Colossal Clean Crawled Corpus)

T5



Text-**t**o-**T**ext **T**ransfer **T**ransformer

T5

- Classificação de Texto (GLUE e SuperGLUE)
- Sumarização (CNN/Daily Mail)
- QA (SQuAD)
- MT (WMT)

T5

- Classificação de Texto (GLUE e SuperGLUE)
- Sumarização (CNN/Daily Mail)
- QA (SQuAD)
- MT (WMT)

Um problema é se o modelo gera “hambúrguer” em vez dos labels aceitos.

T5

- Classificação de Texto (GLUE e SuperGLUE)
- Sumarização (CNN/Daily Mail)
- QA (SQuAD)
- MT (WMT)

Um problema é se o modelo gera “hamburger” em vez dos labels aceitos.

Mas nunca observaram esse comportamento

T5

- Pre-Training
  - Steps = 524.288
  - Max_length = 512
  - Bacth_size = 128 sequencias
  - Tokens = 128 * 512 = 65.536/batch
  - Steps * batch_tokens ~= 34Bi tokens
- Fine-tuning
  - Steps = 262.144
  - Max_length = 512
  - Bacth_size = 128 sequencias
  - Tokens = 128 * 512 = 65.536/batch
  - Learning_rate = 0.001

- Mas e....

RoBERTa

XLEmbeddings

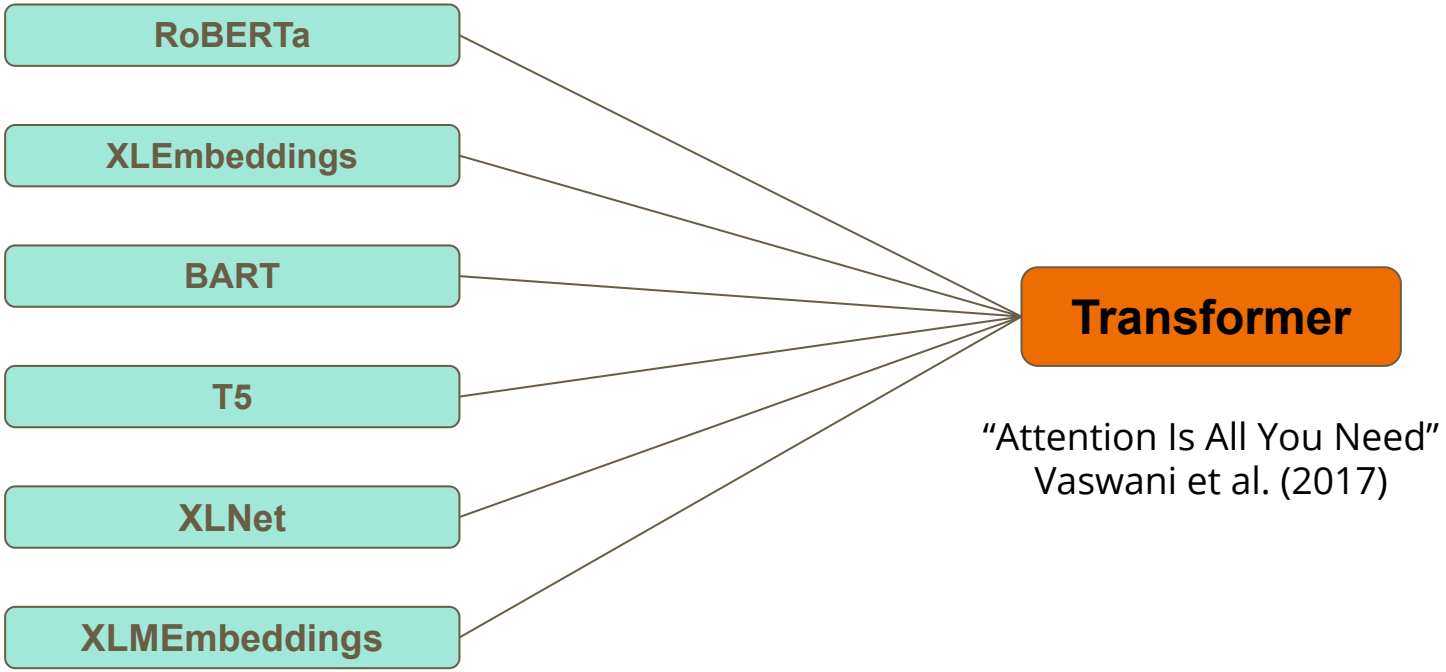
BART

T5

XLNet

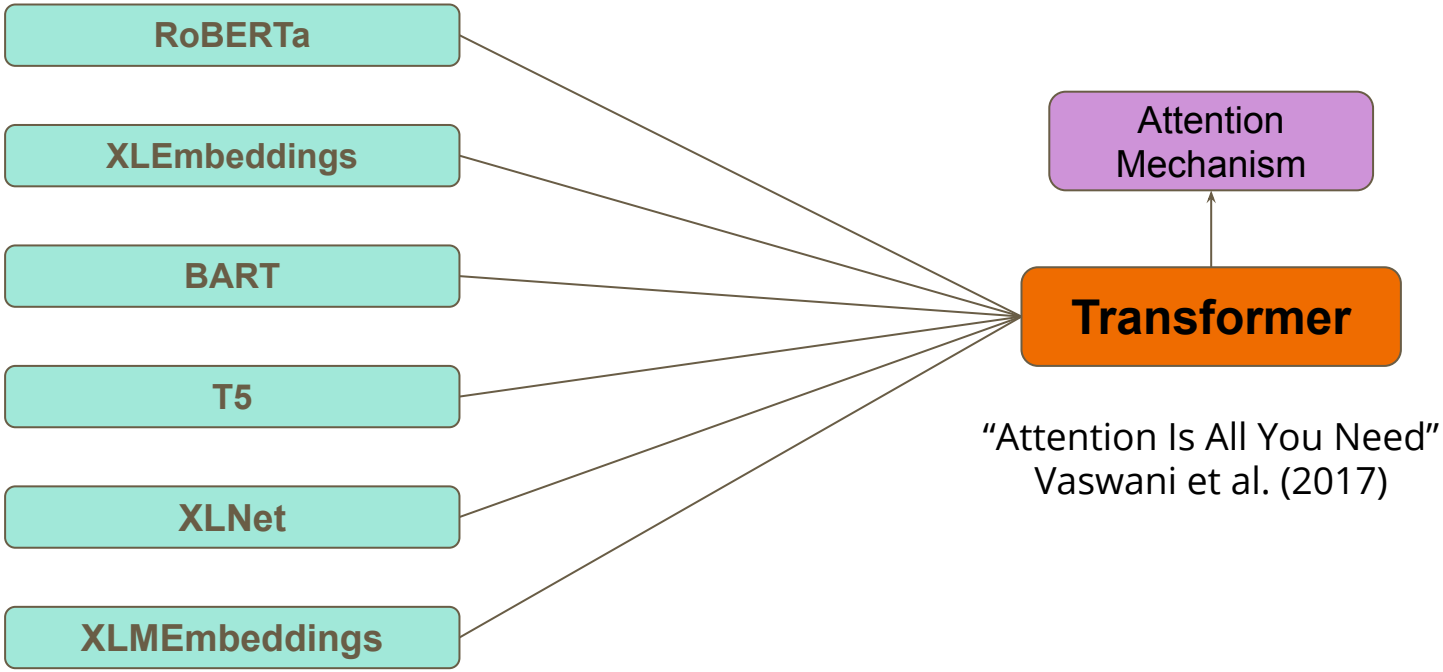
XLMEmbeddings

- Mas e....

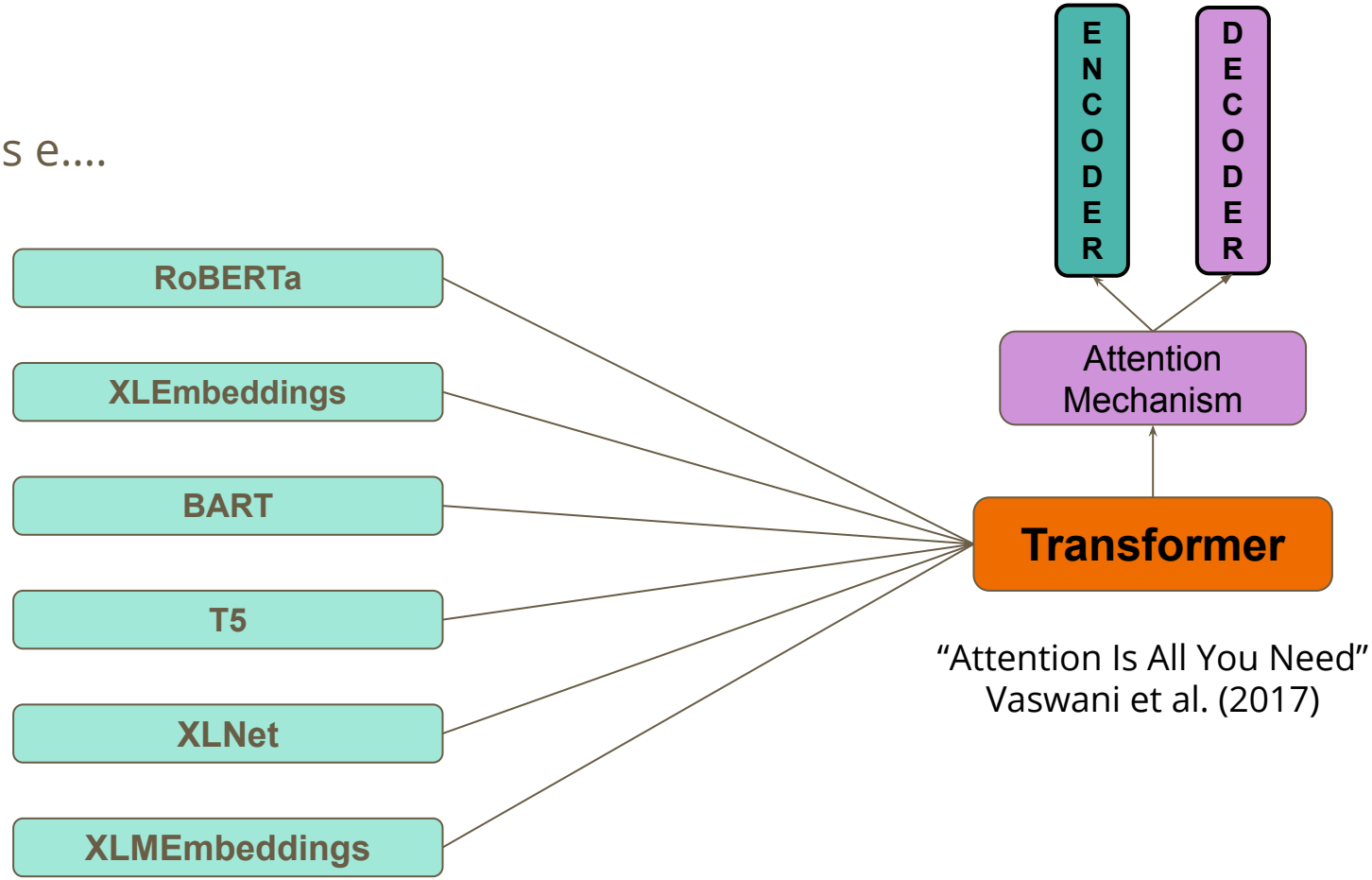


"Attention Is All You Need"
Vaswani et al. (2017)

- Mas e....

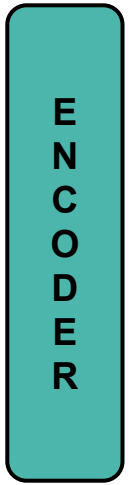

RoBERTa
XLEmbeddings
BART
T5
XLNet
XLMEmbeddings
Attention Mechanism
Transformer
“Attention Is All You Need”
Vaswani et al. (2017)

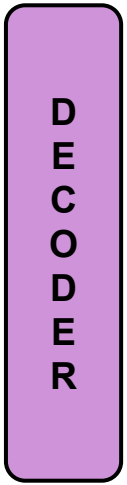
- Mas e....



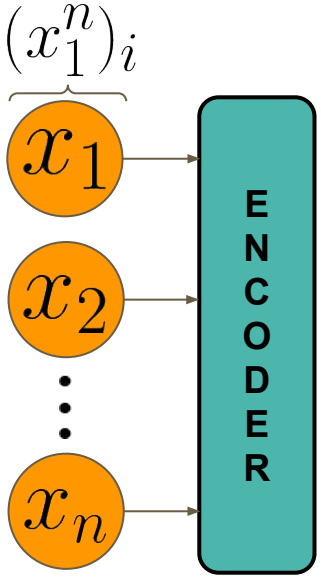
"Attention Is All You Need"
Vaswani et al. (2017)

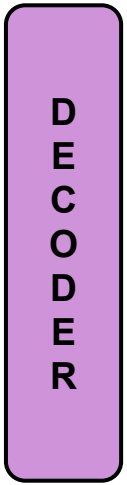
Attention Mechanism



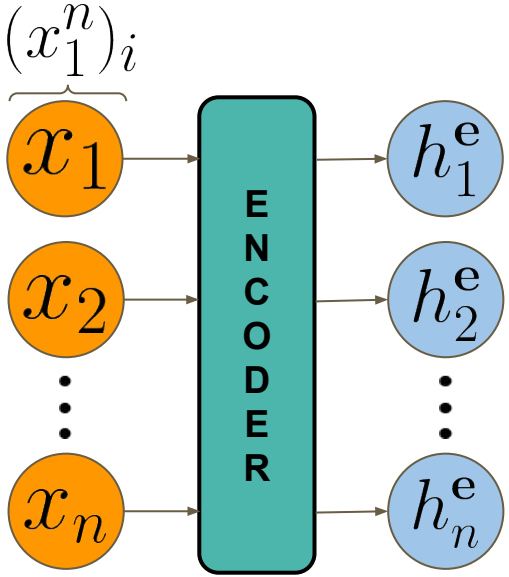


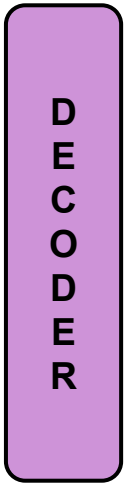
Attention Mechanism

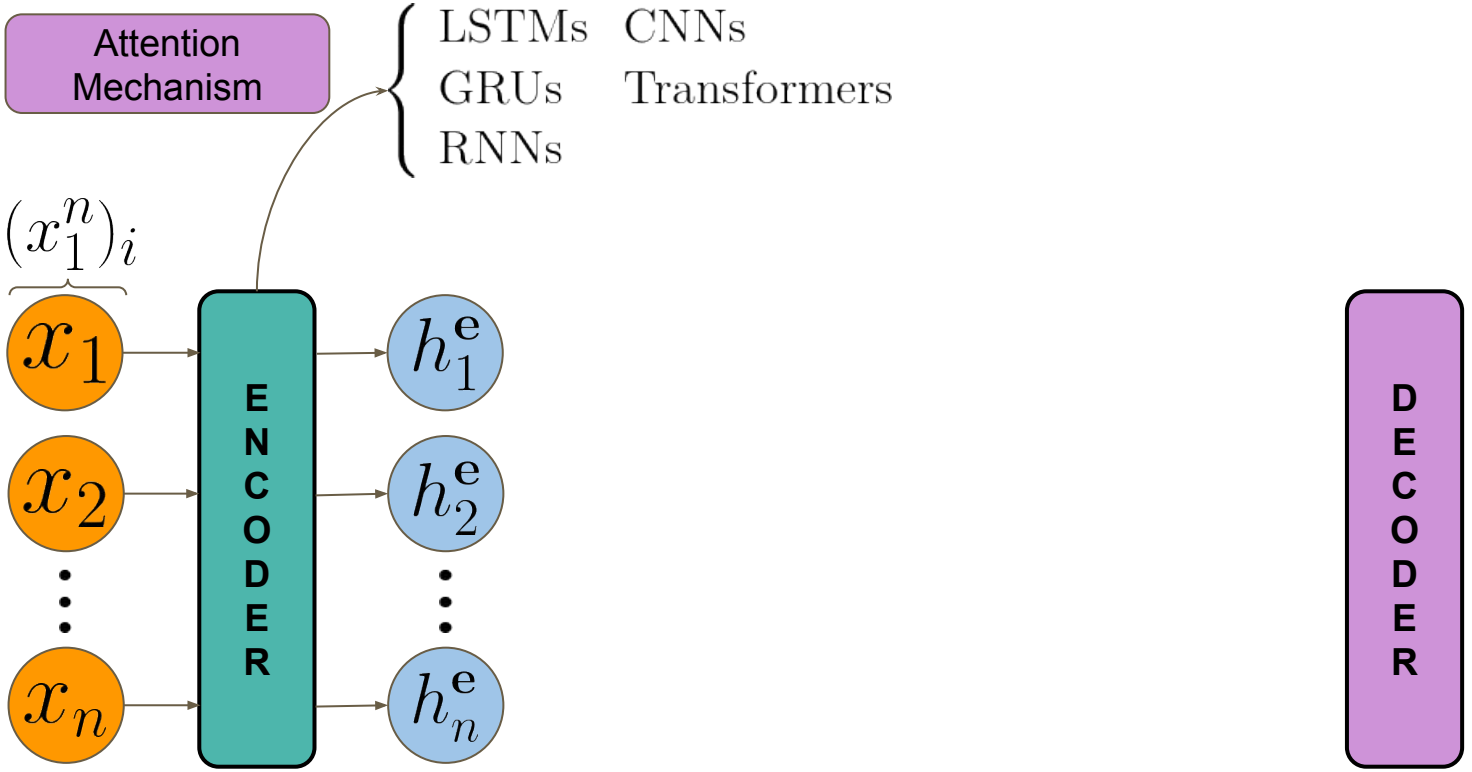


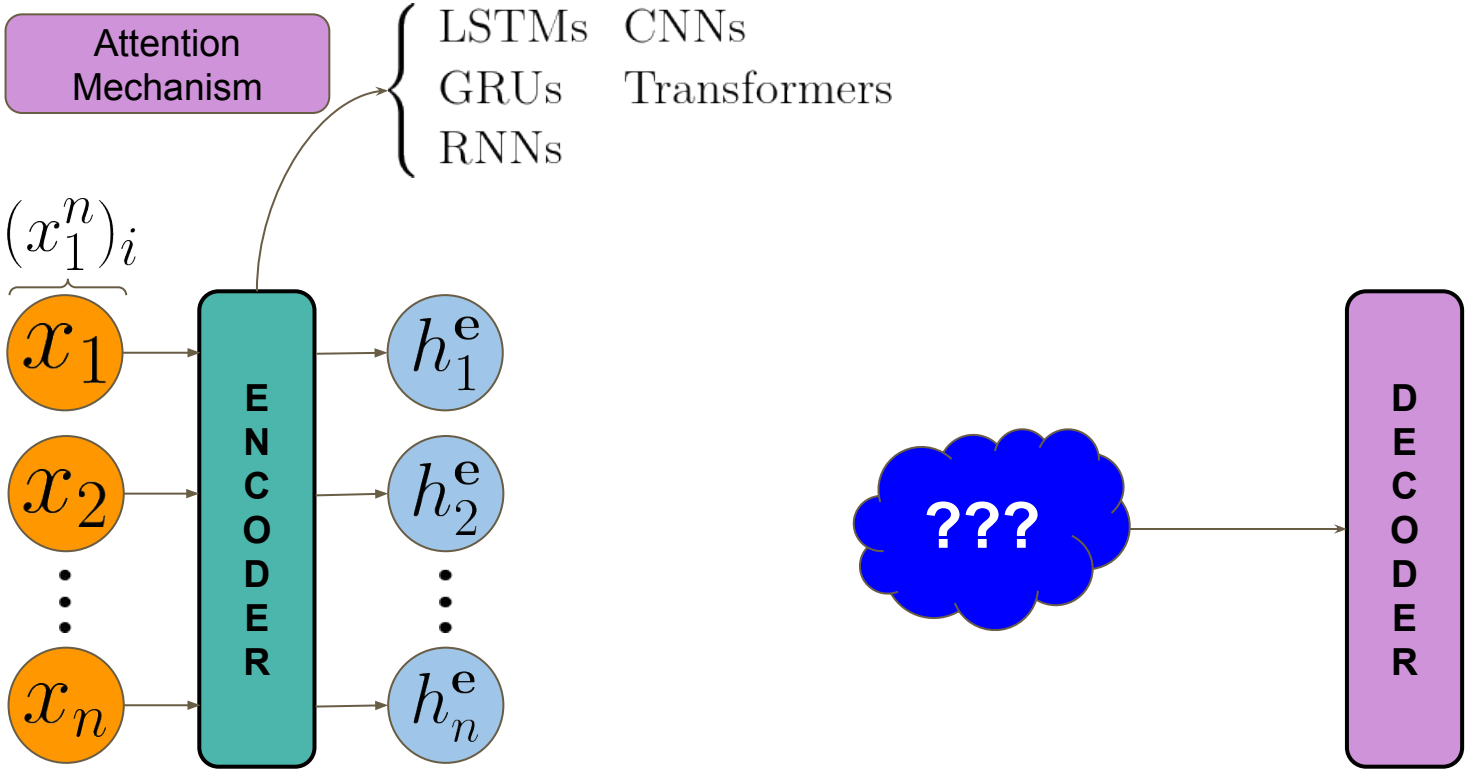


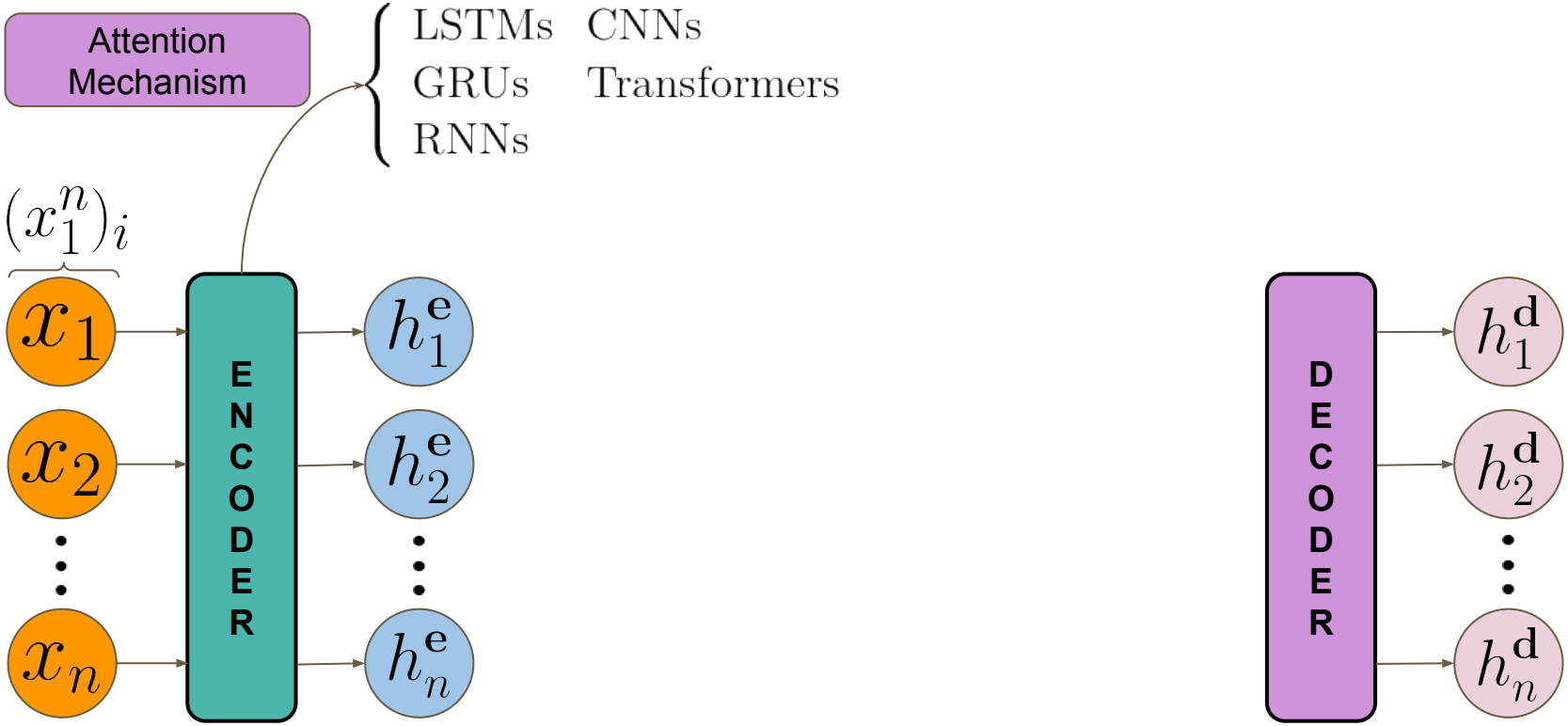
Attention Mechanism

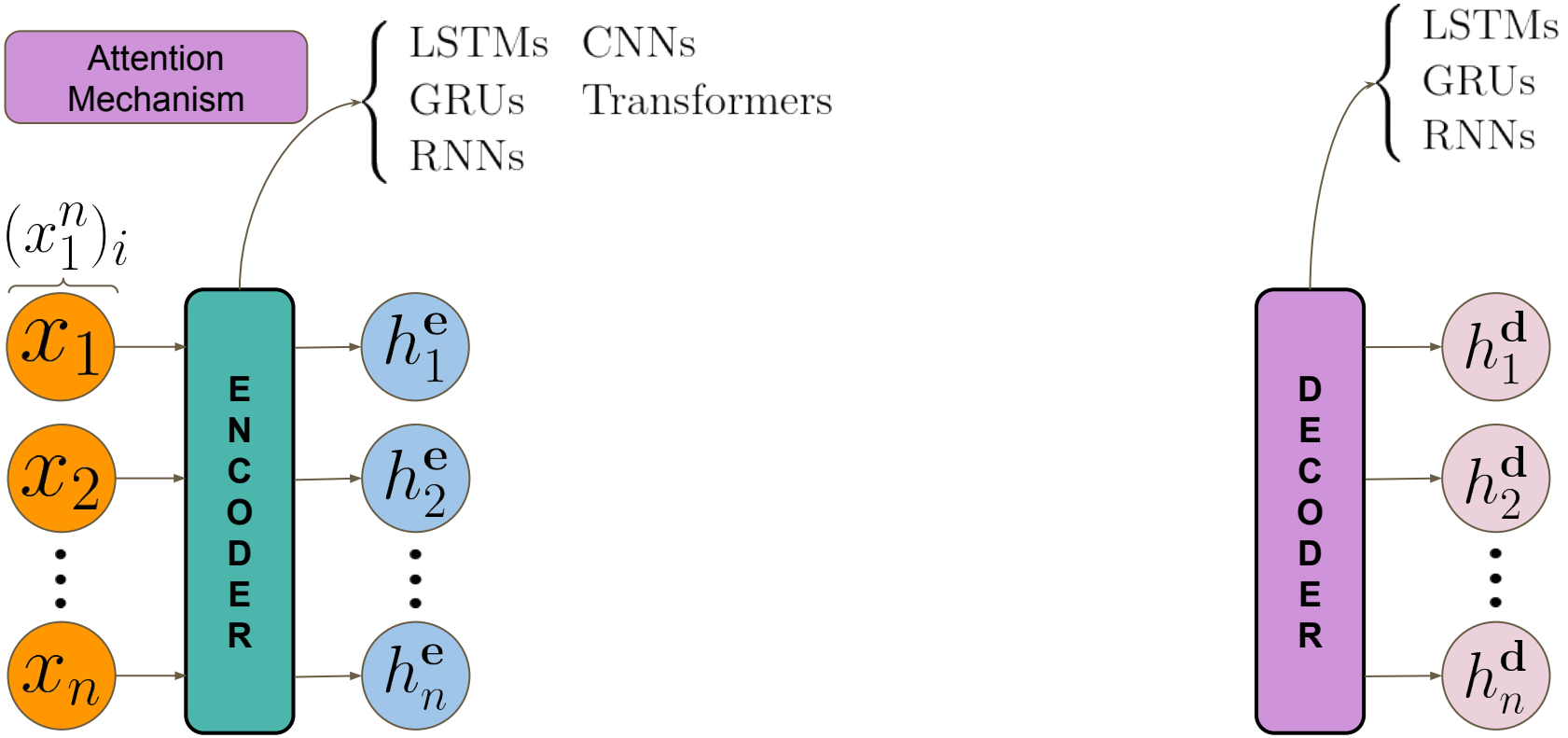


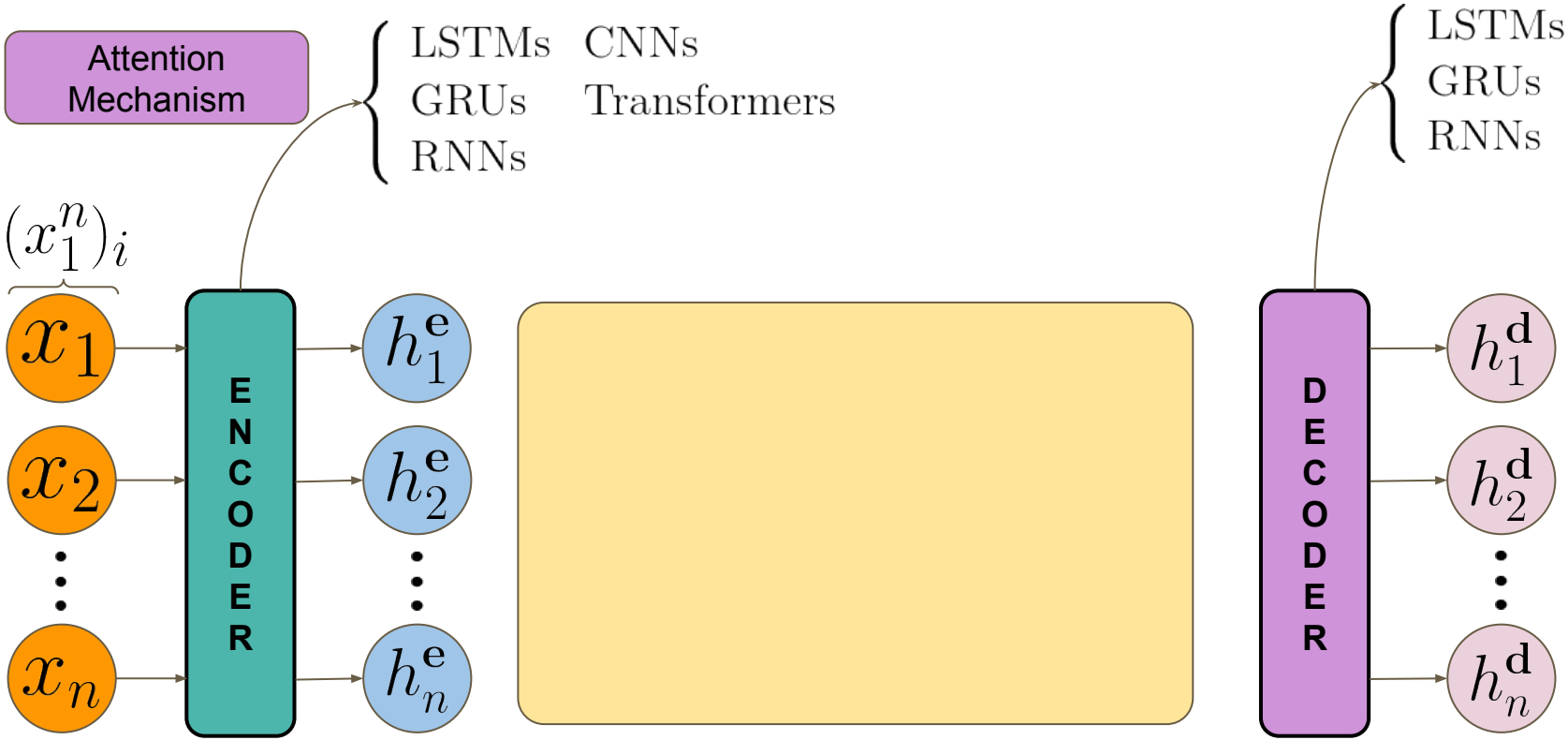


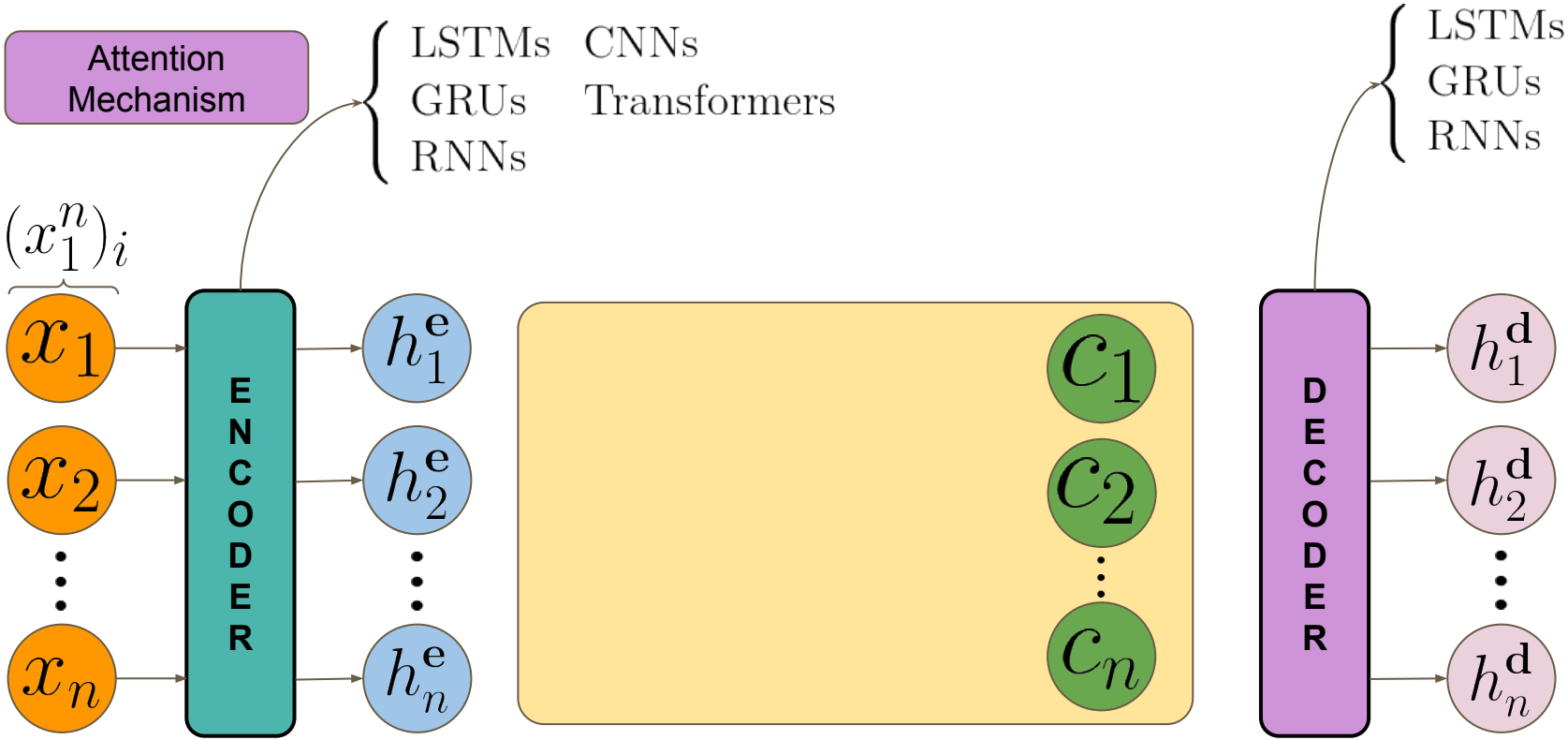

Attention Mechanism
LSTMs CNNs
GRUs Transformers
RNNs
$(x_1^n)_i$
x_1
x_2
x_n
ENCODER
h_1^e
h_2^e
h_n^e
DECODER

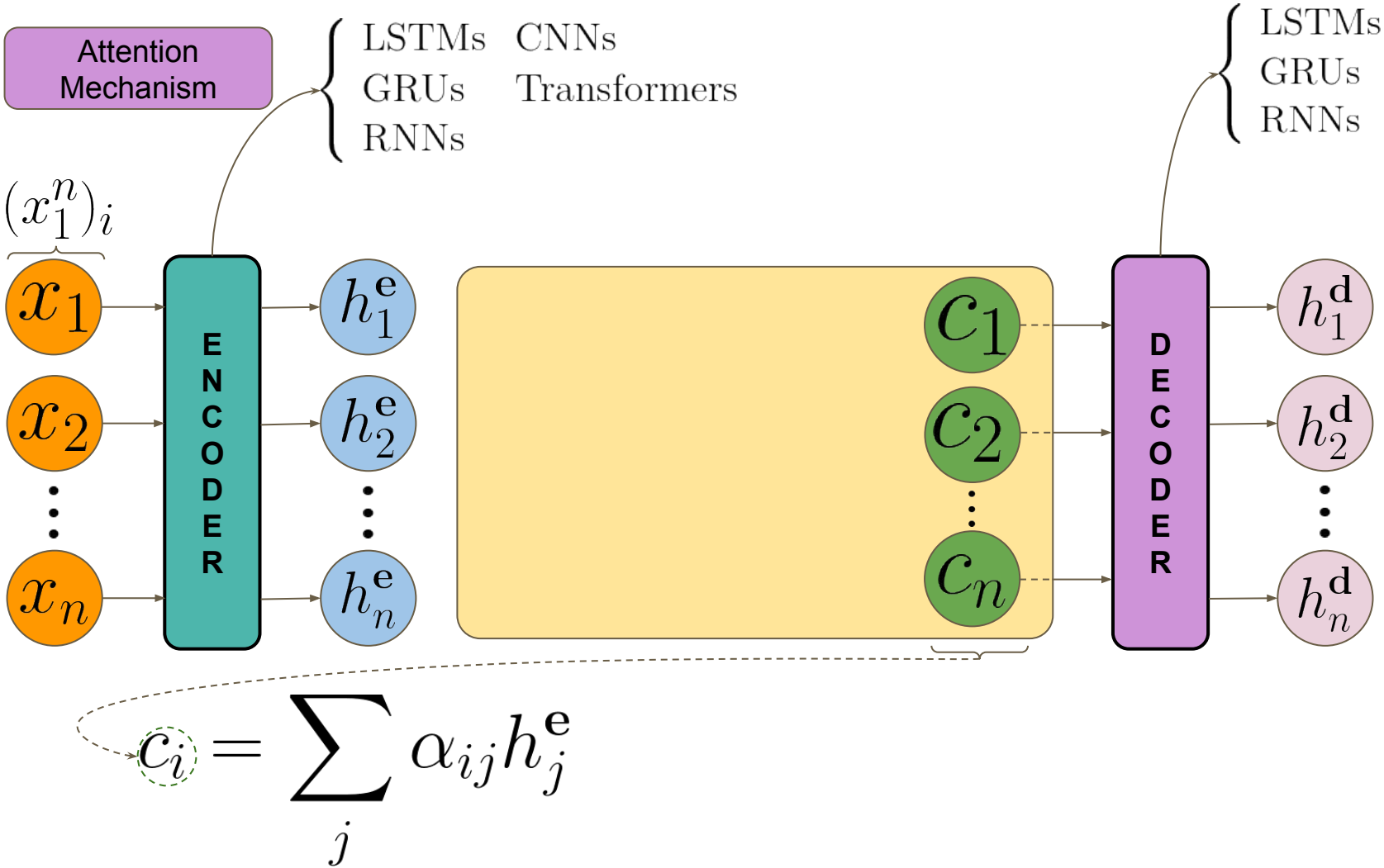

Attention Mechanism
LSTMs CNNs
GRUs Transformers
RNNs
$(x_1^n)_i$
x_1
x_2
x_n
ENCODER
h_1^e
h_2^e
h_n^e
???
DECODER

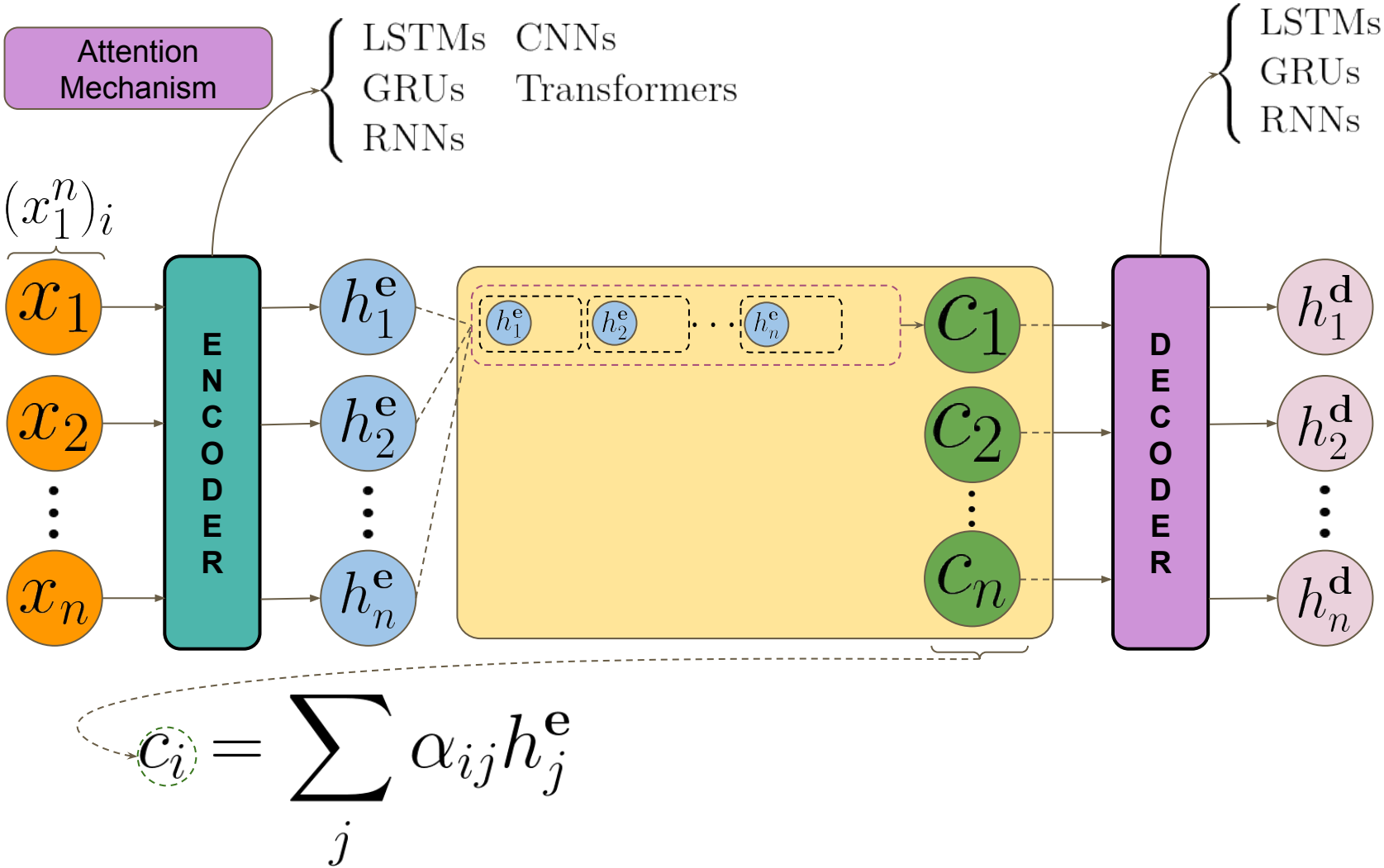

Attention Mechanism
LSTMs CNNs
GRUs Transformers
RNNs
$(x_1^n)_i$
x_1
x_2
x_n
ENCODER
h_1^e
h_2^e
h_n^e
DECODER
h_1^d
h_2^d
h_n^d

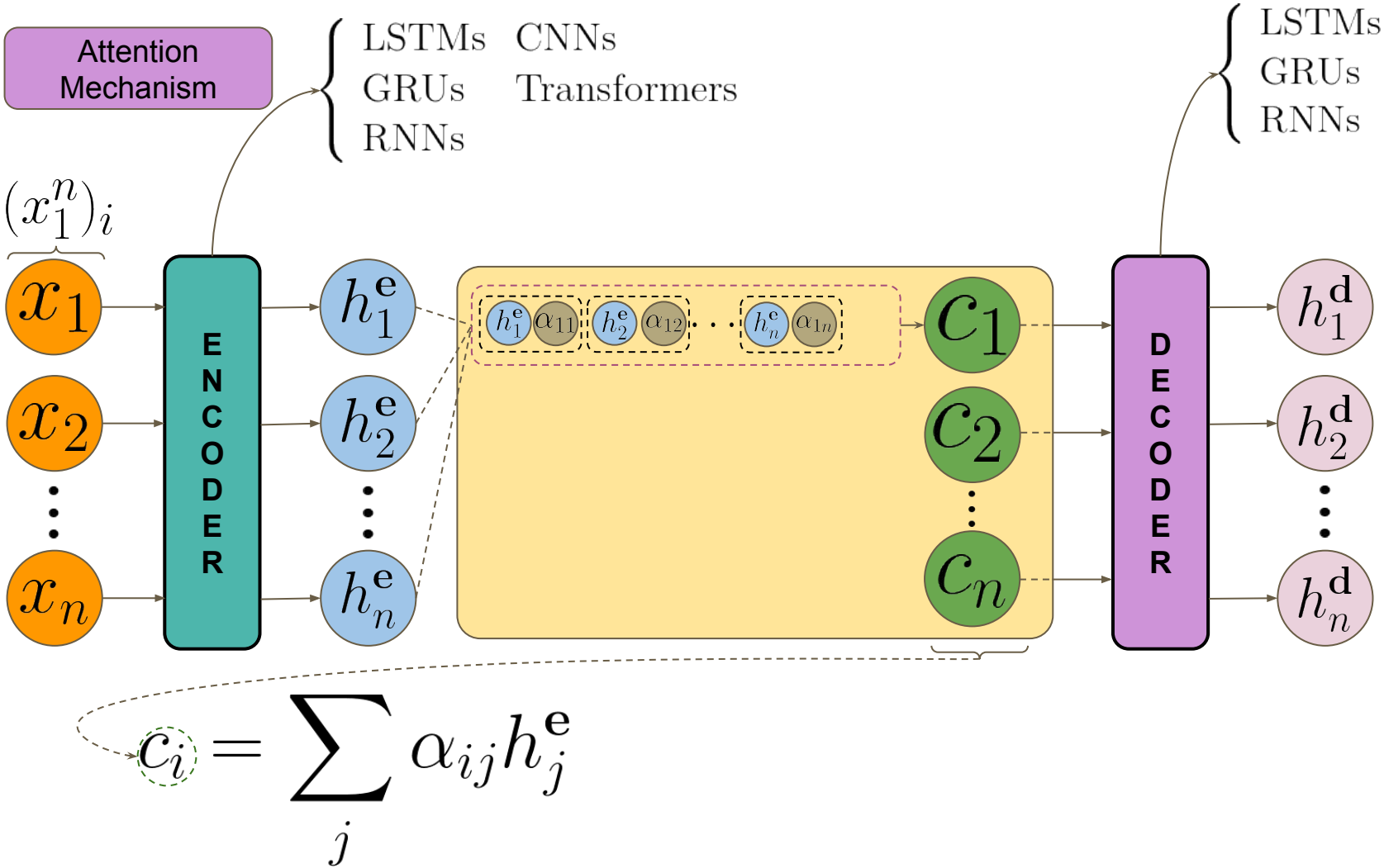

Attention Mechanism
LSTMs CNNs
GRUs Transformers
RNNs
$(x_1^n)_i$
x_1
x_2
x_n
ENCODER
h_1^e
h_2^e
h_n^e
LSTMs
GRUs
RNNs
DECODER
h_1^d
h_2^d
h_n^d

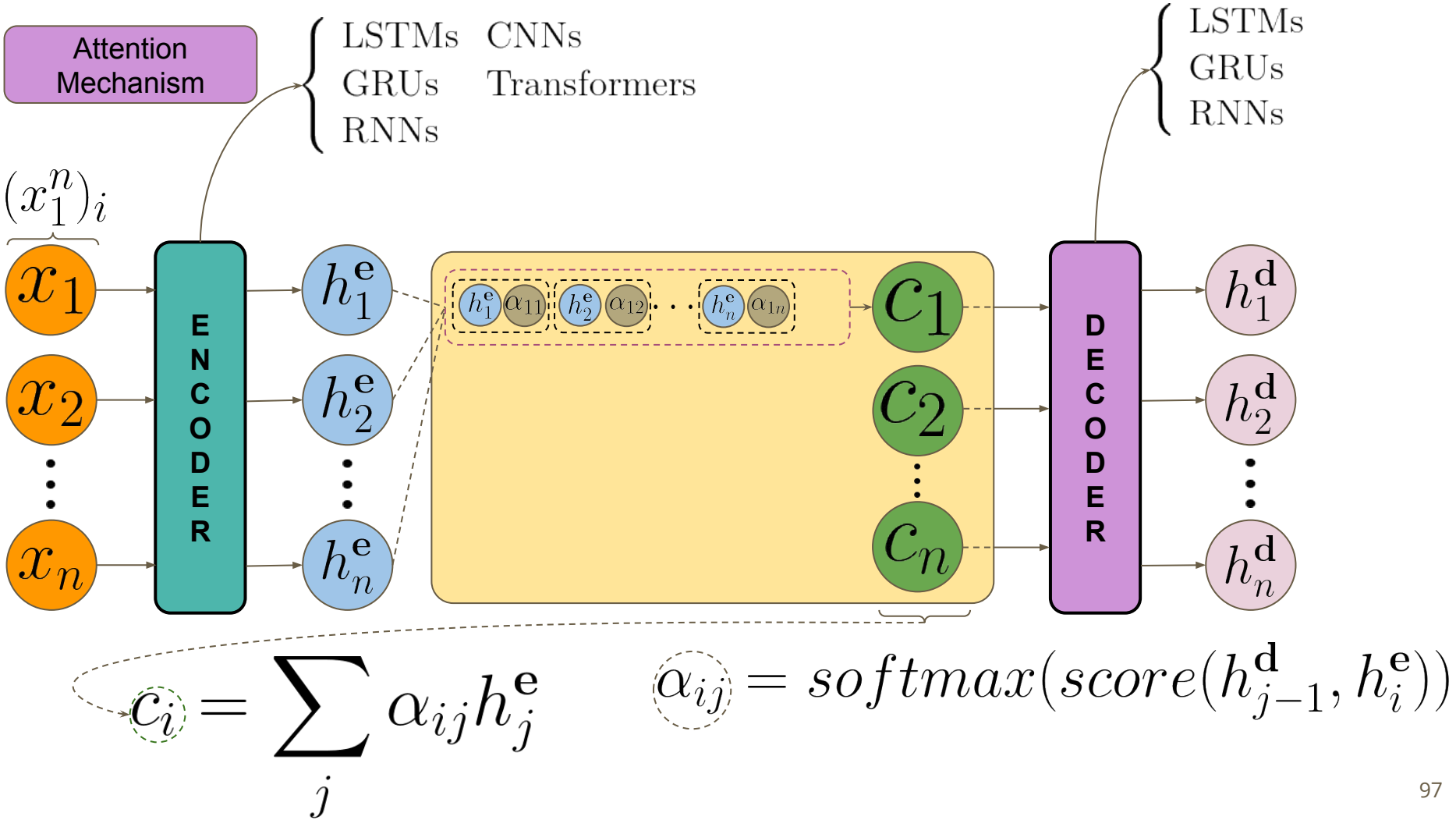

Attention Mechanism
LSTMs CNNs
GRUs Transformers
RNNs
LSTMs
GRUs
RNNs
$(x_1^n)_i$
x_1
x_2
x_n
ENCODER
h_1^e
h_2^e
h_n^e
DECODER
h_1^d
h_2^d
h_n^d


Attention Mechanism
LSTMs CNNs
GRUs Transformers
RNNs
LSTMs
GRUs
RNNs
$(x_1^n)_i$
x_1
x_2
x_n
ENCODER
h_1^e
h_2^e
h_n^e
c_1
c_2
c_n
DECODER
h_1^d
h_2^d
h_n^d


Attention Mechanism
LSTMs CNNs
GRUs Transformers
RNNs
LSTMs
GRUs
RNNs
$(x_1^n)_i$
x_1
x_2
x_n
ENCODER
h_1^e
h_2^e
h_n^e
c_1
c_2
c_n
DECODER
h_1^d
h_2^d
h_n^d
$c_i = \sum_j \alpha_{ij} h_j^e$


Attention Mechanism
LSTMs CNNs
GRUs Transformers
RNNs
LSTMs
GRUs
RNNs
$(x_1^n)_i$
x_1
x_2
x_n
ENCODER
h_1^e
h_2^e
h_n^e
c_1
c_2
c_n
DECODER
h_1^d
h_2^d
h_n^d
$c_i = \sum_j \alpha_{ij} h_j^e$

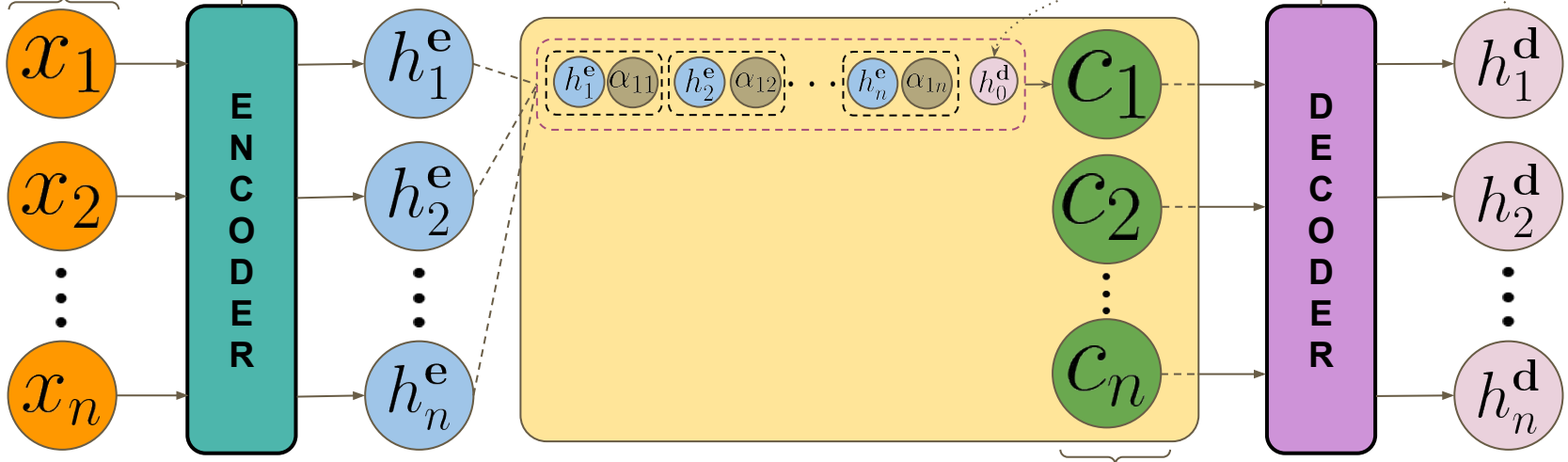

Attention Mechanism
LSTMs CNNs
GRUs Transformers
RNNs
LSTMs
GRUs
RNNs
$(x_1^n)_i$
x_1
x_2
x_n
ENCODER
h_1^e
h_2^e
h_n^e
h_1^e α_{11} h_2^e α_{12} h_n^e α_{1n}
c_1
c_2
c_n
DECODER
h_1^d
h_2^d
h_n^d
$c_i = \sum_j \alpha_{ij} h_j^e$


Attention Mechanism
LSTMs CNNs
GRUs Transformers
RNNs
LSTMs
GRUs
RNNs
$(x_1^n)_i$
x_1
x_2
x_n
ENCODER
h_1^e
h_2^e
h_n^e
h_1^e α_{11} h_2^e α_{12} h_n^e α_{1n}
c_1
c_2
c_n
DECODER
h_1^d
h_2^d
h_n^d
$c_i = \sum_j \alpha_{ij} h_j^e$
$\alpha_{ij} = softmax(score(h_{j-1}^d, h_i^e))$

Attention Mechanism

LSTMs CNNs
GRUs Transformers
RNNs

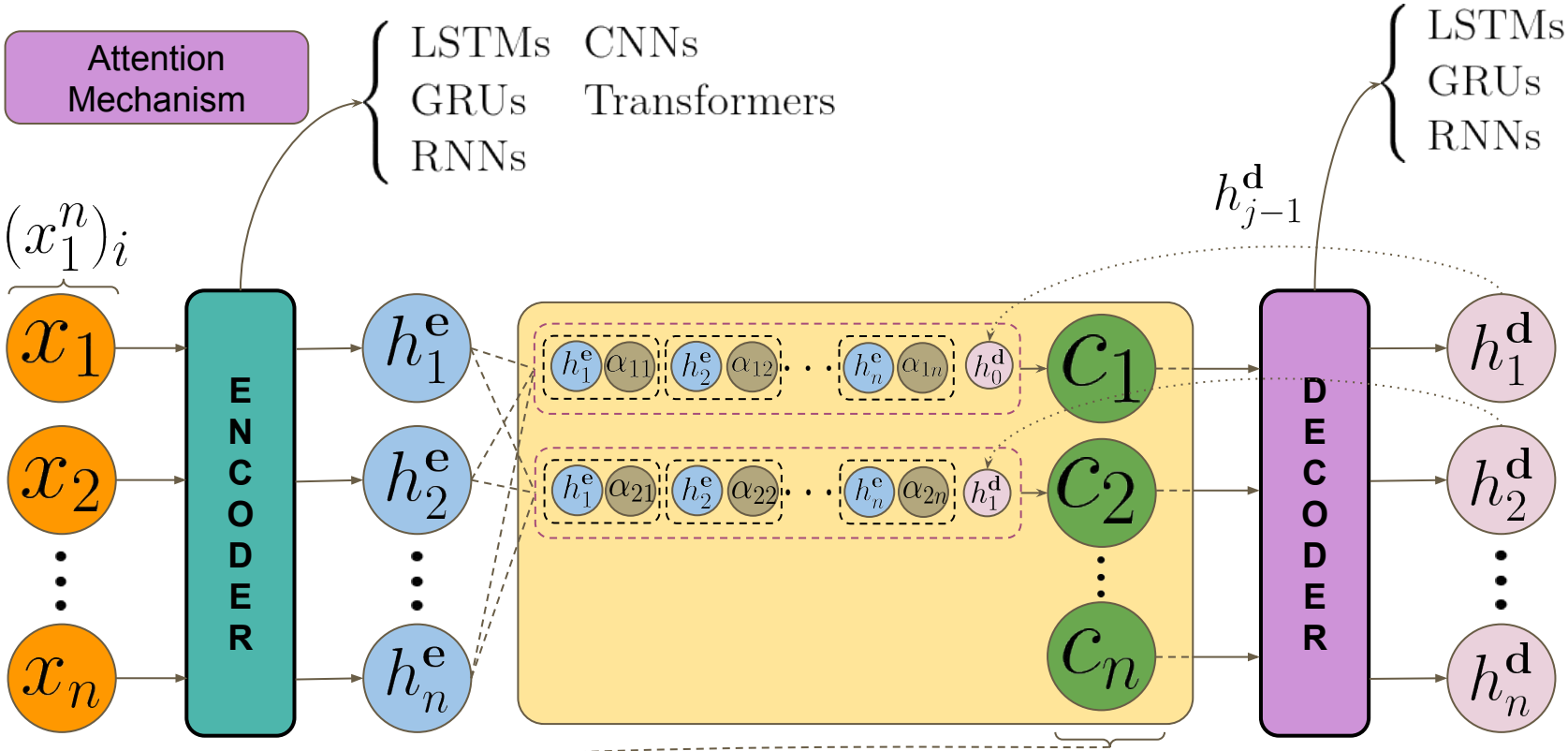
LSTMs
GRUs
RNNs

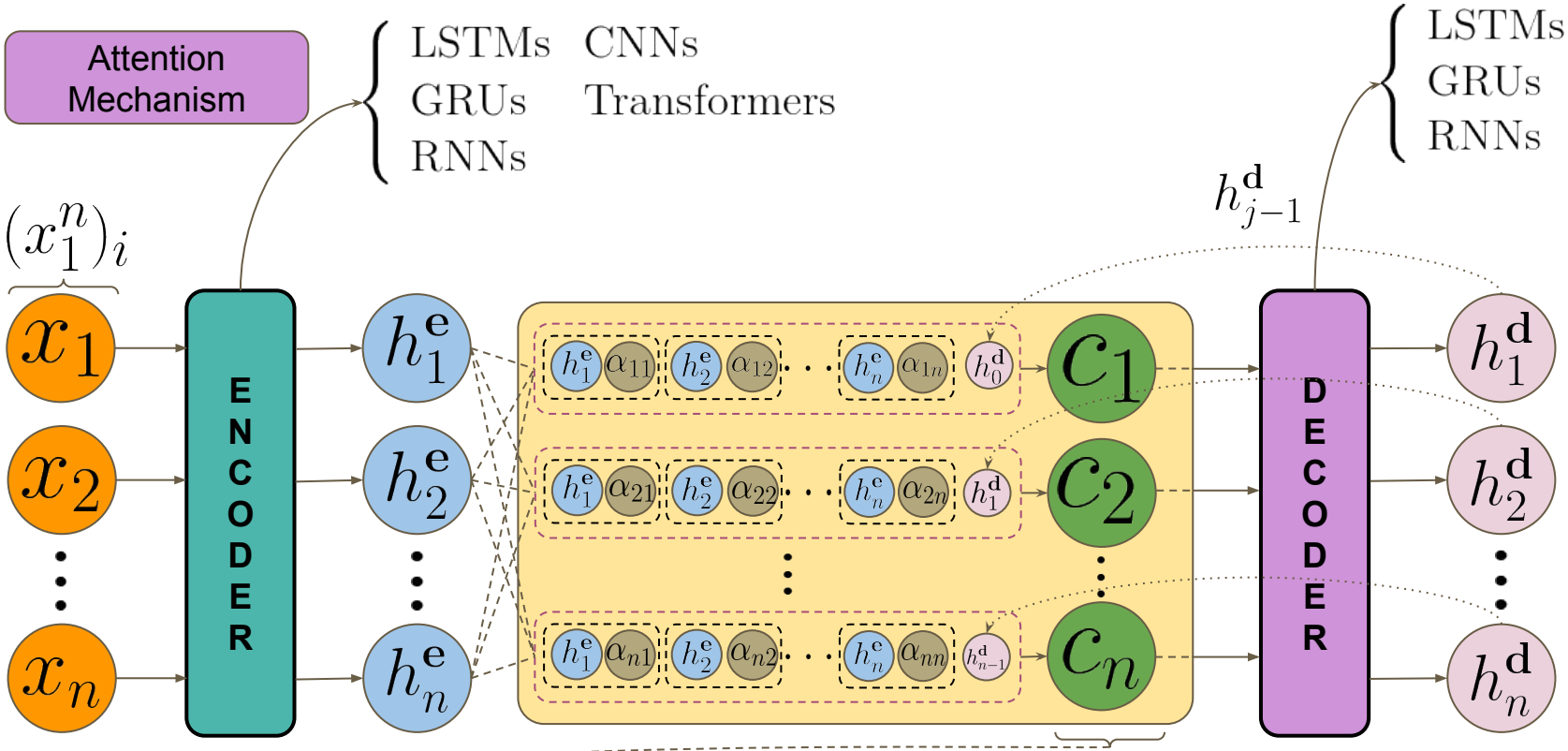


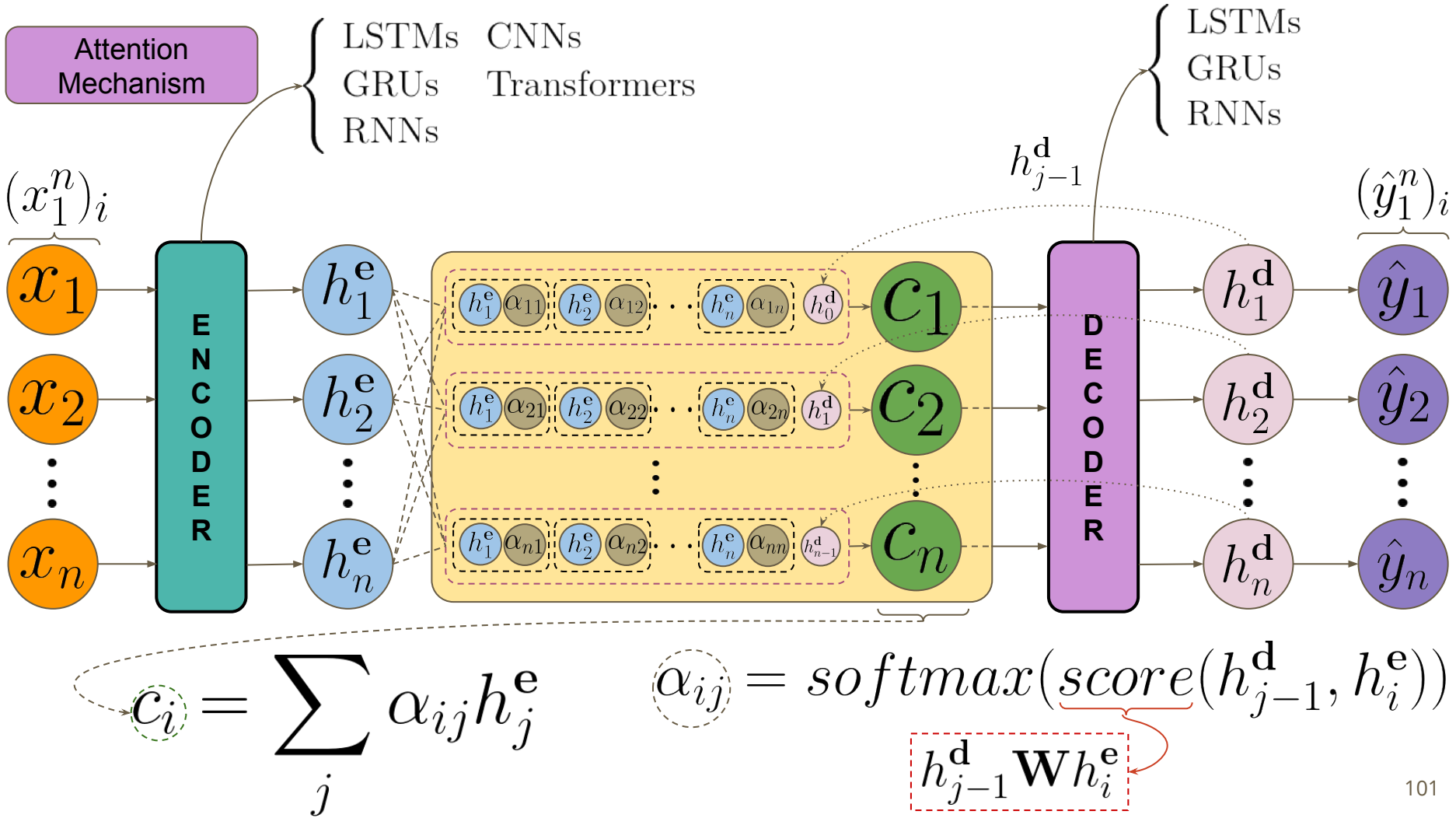
$$c_i = \sum_j \alpha_{ij} h_j^e$$

$$\alpha_{ij} = softmax(score(h_{j-1}^d, h_i^e))$$

$$h_{j-1}^d \mathbf{W} h_i^e$$


Attention Mechanism
LSTMs CNNs
GRUs Transformers
RNNs
LSTMs
GRUs
RNNs
$(x_1^n)_i$
$h_{j-1}^{\mathbf{d}}$
x_1
x_2
x_n
ENCODER
$h_1^{\mathbf{e}}$
$h_2^{\mathbf{e}}$
$h_n^{\mathbf{e}}$
$h_1^{\mathbf{e}}\ \alpha_{11}\ h_2^{\mathbf{e}}\ \alpha_{12}\ \cdots\ h_n^{\mathbf{e}}\ \alpha_{1n}\ h_0^{\mathbf{d}}$
$h_1^{\mathbf{e}}\ \alpha_{21}\ h_2^{\mathbf{e}}\ \alpha_{22}\ \cdots\ h_n^{\mathbf{e}}\ \alpha_{2n}\ h_1^{\mathbf{d}}$
c_1
c_2
c_n
DECODER
$h_1^{\mathbf{d}}$
$h_2^{\mathbf{d}}$
$h_n^{\mathbf{d}}$
$c_i = \sum_j \alpha_{ij} h_j^{\mathbf{e}}$
$\alpha_{ij} = softmax(score(h_{j-1}^{\mathbf{d}}, h_i^{\mathbf{e}}))$
$h_{j-1}^{\mathbf{d}} \mathbf{W} h_i^{\mathbf{e}}$


Attention Mechanism
LSTMs CNNs
GRUs Transformers
RNNs
LSTMs
GRUs
RNNs
$(x_1^n)_i$
$h_{j-1}^{\mathbf{d}}$
x_1
x_2
x_n
ENCODER
$h_1^{\mathbf{e}}$
$h_2^{\mathbf{e}}$
$h_n^{\mathbf{e}}$
$h_1^{\mathbf{e}}\ \alpha_{11}\ h_2^{\mathbf{e}}\ \alpha_{12}\ \cdots\ h_n^{\mathbf{e}}\ \alpha_{1n}\ h_0^{\mathbf{d}}$
$h_1^{\mathbf{e}}\ \alpha_{21}\ h_2^{\mathbf{e}}\ \alpha_{22}\ \cdots\ h_n^{\mathbf{e}}\ \alpha_{2n}\ h_1^{\mathbf{d}}$
$h_1^{\mathbf{e}}\ \alpha_{n1}\ h_2^{\mathbf{e}}\ \alpha_{n2}\ \cdots\ h_n^{\mathbf{e}}\ \alpha_{nn}\ h_{n-1}^{\mathbf{d}}$
c_1
c_2
c_n
DECODER
$h_1^{\mathbf{d}}$
$h_2^{\mathbf{d}}$
$h_n^{\mathbf{d}}$
$c_i = \sum_j \alpha_{ij} h_j^{\mathbf{e}}$
$\alpha_{ij} = softmax(score(h_{j-1}^{\mathbf{d}}, h_i^{\mathbf{e}}))$
$h_{j-1}^{\mathbf{d}} \mathbf{W} h_i^{\mathbf{e}}$

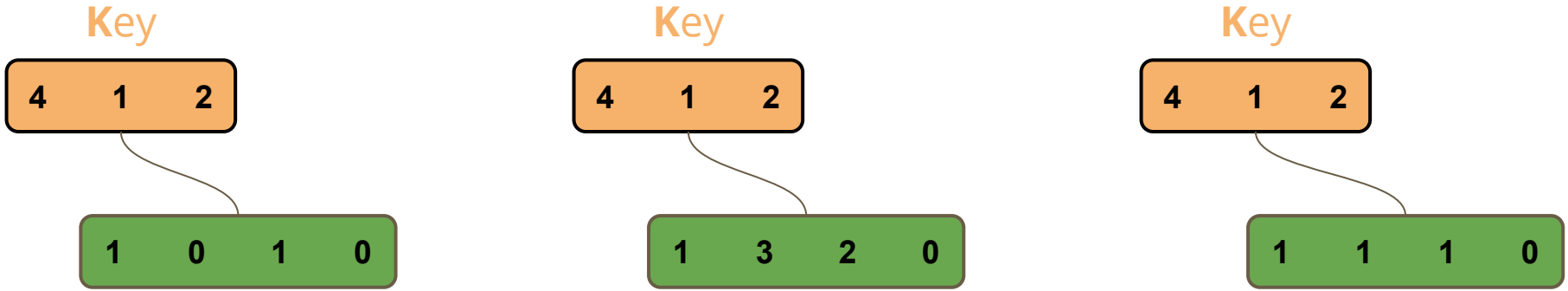

Attention Mechanism
LSTMs CNNs
GRUs Transformers
RNNs
LSTMs
GRUs
RNNs
$(x_1^n)_i$
$(\hat{y}_1^n)_i$
$h_{j-1}^{\mathbf{d}}$
x_1
x_2
x_n
ENCODER
$h_1^{\mathbf{e}}$
$h_2^{\mathbf{e}}$
$h_n^{\mathbf{e}}$
$h_1^{\mathbf{e}}\ \alpha_{11}\ h_2^{\mathbf{e}}\ \alpha_{12}\ \cdots\ h_n^{\mathbf{e}}\ \alpha_{1n}\ h_0^{\mathbf{d}}$
$h_1^{\mathbf{e}}\ \alpha_{21}\ h_2^{\mathbf{e}}\ \alpha_{22}\ \cdots\ h_n^{\mathbf{e}}\ \alpha_{2n}\ h_1^{\mathbf{d}}$
$h_1^{\mathbf{e}}\ \alpha_{n1}\ h_2^{\mathbf{e}}\ \alpha_{n2}\ \cdots\ h_n^{\mathbf{e}}\ \alpha_{nn}\ h_{n-1}^{\mathbf{d}}$
c_1
c_2
c_n
DECODER
$h_1^{\mathbf{d}}$
$h_2^{\mathbf{d}}$
$h_n^{\mathbf{d}}$
$\hat{y}_1$
$\hat{y}_2$
$\hat{y}_n$
$c_i = \sum_j \alpha_{ij} h_j^{\mathbf{e}}$
$\alpha_{ij} = softmax(score(h_{j-1}^{\mathbf{d}}, h_i^{\mathbf{e}}))$
$h_{j-1}^{\mathbf{d}} \mathbf{W} h_i^{\mathbf{e}}$

Attention Mechanism

$$\text{Att}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = Softmax(\frac{1}{\sqrt{d_k}}\mathbf{Q}\mathbf{K}^T)\mathbf{V}$$

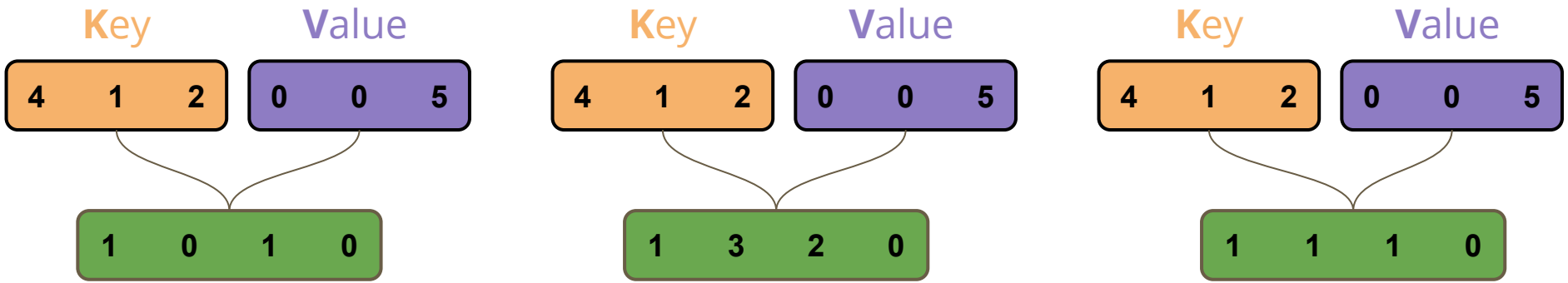
Attention Mechanism

$$\text{Att}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = Softmax(\frac{1}{\sqrt{d_k}}\mathbf{Q}\mathbf{K}^T)\mathbf{V}$$



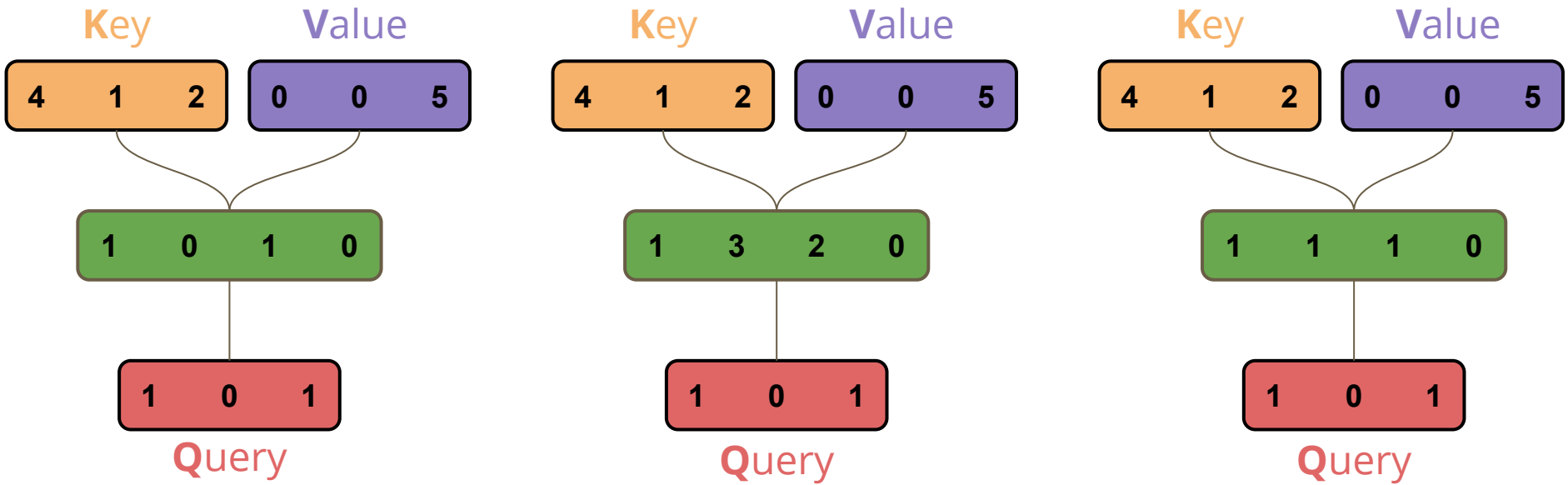
Attention Mechanism

$$\text{Att}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = Softmax(\frac{1}{\sqrt{d_k}}\mathbf{Q}\mathbf{K}^T)\mathbf{V}$$



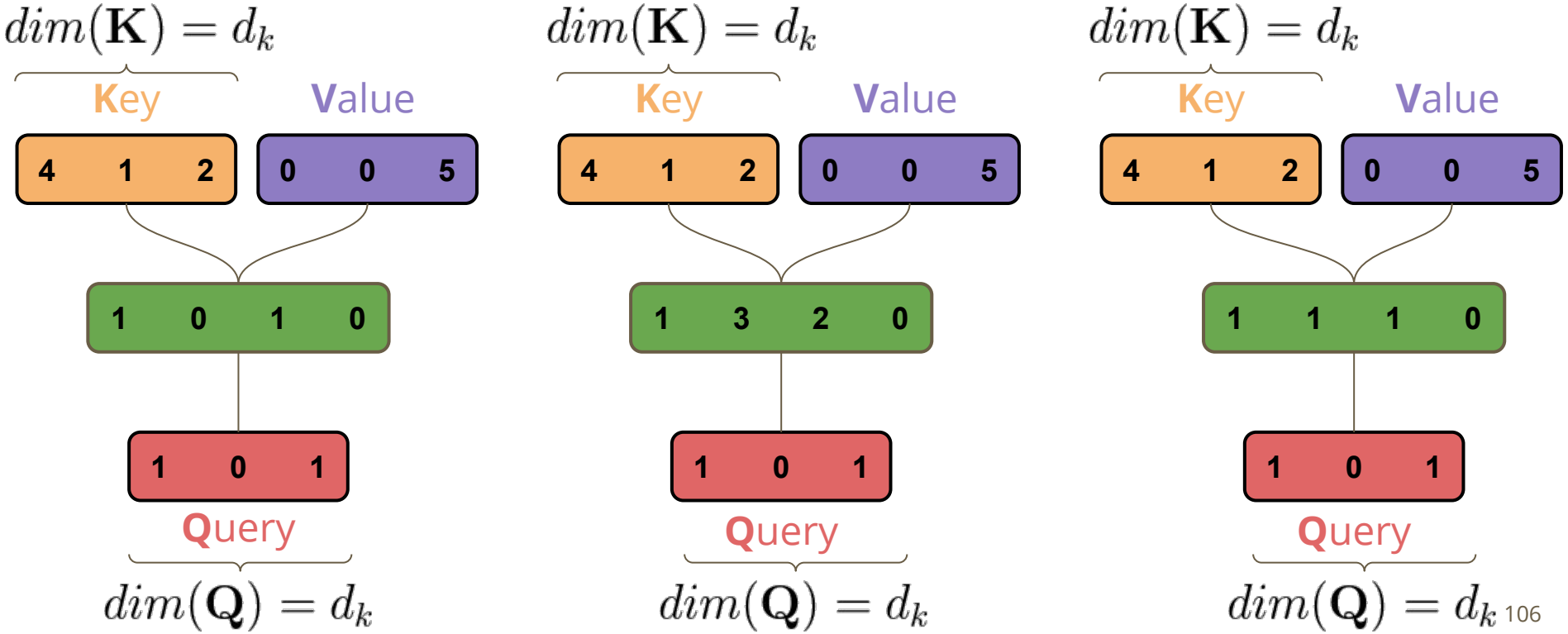
Attention Mechanism

$$\text{Att}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = Softmax(\frac{1}{\sqrt{d_k}}\mathbf{Q}\mathbf{K}^T)\mathbf{V}$$



Attention Mechanism

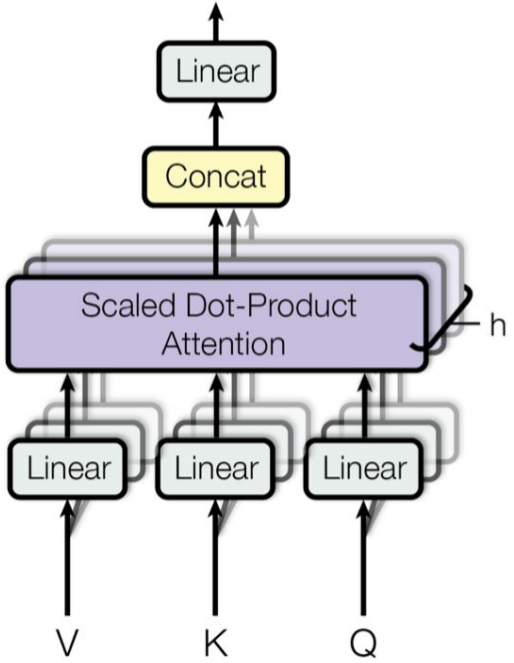
$$\text{Att}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = Softmax(\frac{1}{\sqrt{d_k}}\mathbf{Q}\mathbf{K}^T)\mathbf{V}$$



$$\text{Att}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = Softmax(\frac{1}{\sqrt{d_k}}\mathbf{Q}\mathbf{K}^T)\mathbf{V}$$

$$\text{MultiHead}(\mathbf{Q},\mathbf{K},\mathbf{V}) = Concat(head_1, head_2, ..., head_h)\mathbf{W}^O$$

$$head_i = Att(\mathbf{Q}\mathbf{W}_i^Q, \mathbf{K}\mathbf{W}_i^K, \mathbf{V}\mathbf{W}_i^V)$$

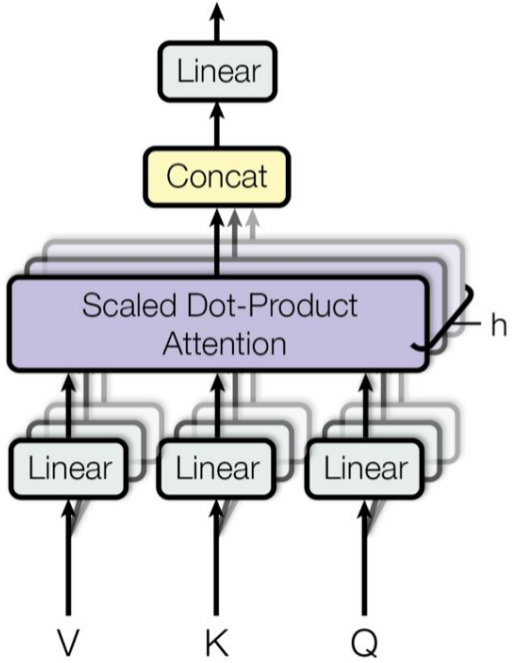


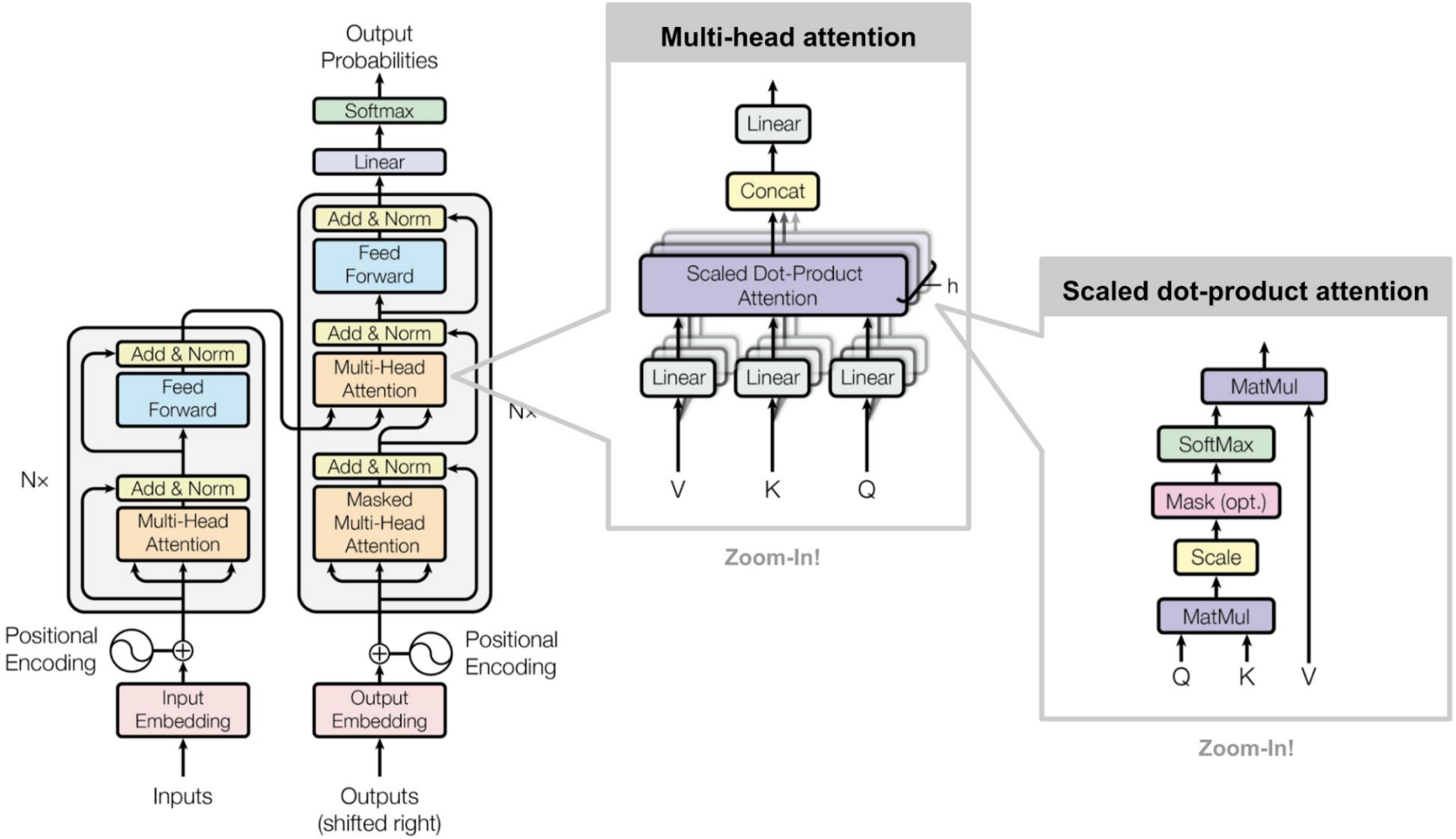
$$\text{Att}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = Softmax(\frac{1}{\sqrt{d_k}}\mathbf{Q}\mathbf{K}^T)\mathbf{V}$$

$$\text{MultiHead}(\mathbf{Q},\mathbf{K},\mathbf{V}) = Concat(head_1, head_2, ..., head_h)\mathbf{W}^O$$

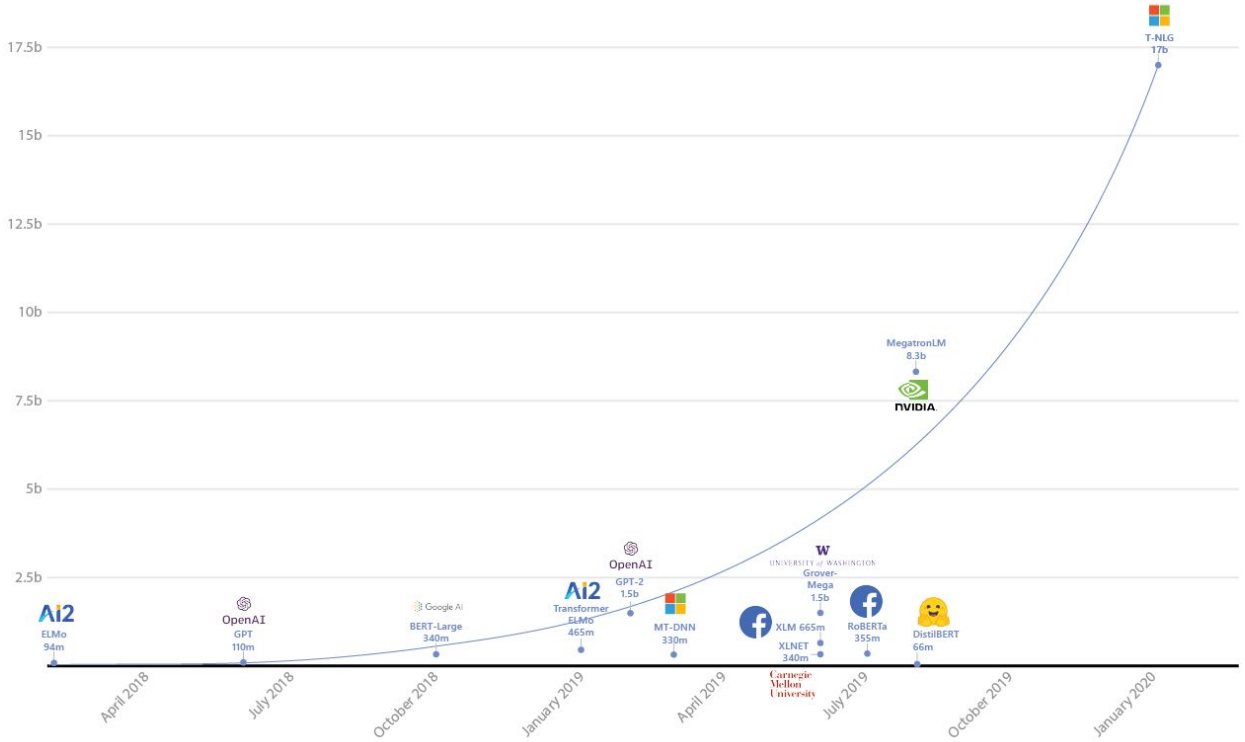
$$head_i = Att(\mathbf{Q}\mathbf{W}_i^Q, \mathbf{K}\mathbf{W}_i^K, \mathbf{V}\mathbf{W}_i^V)$$

Matrizes de parâmetros a serem aprendidas




Output Probabilities
Softmax
Linear
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Add & Norm
Masked Multi-Head Attention
Nx
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Nx
Positional Encoding
Positional Encoding
Input Embedding
Output Embedding
Inputs
Outputs (shifted right)
Multi-head attention
Linear
Concat
Scaled Dot-Product Attention
h
Linear
Linear
Linear
V
K
Q
Zoom-In!
Scaled dot-product attention
MatMul
SoftMax
Mask (opt.)
Scale
MatMul
Q
K
V
Zoom-In!

Curiosidade



- Dicas de Leitura
  - Jurafsky
  - https://lilianweng.github.io/lil-log/2018/06/24/attention-attention.html#summary
  - Jay Alammar - http://jalammar.github.io/illustrated-transformer/
  - “Attention is all you need!” - http://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf
  - https://lilianweng.github.io/lil-log/2018/06/24/attention-attention.html#summary

Modelos para a língua portuguesa

Word Embeddings para o Português

LX - Center Portugal

1,7 Bilhões de Tokens

Word2Vec

LX Center - Disponível em:
http://lxcenter.di.fc.ul.pt/datasets/en/index.html

NILC Embeddings

1,3 Bilhões de Tokens

Word2Vec **FastText**

Wang2vec **Glove**

NILC Embeddings - Disponível em:
http://nilc.icmc.usp.br/embeddings

PLN-PUCRS

4,9 Bilhões de Tokens

Word2Vec **FastText**

PLN-PUCRS - Disponível em:
https://github.com/jneto04/ner-pt

Flair Embeddings para o Português

Charles University FlairEL

0,9 Bilhões de Tokens

Forward	*2.78*
Backward	*2.81*

Eric Lief - Disponível em:
https://github.com/zalandoresearch/flair

PUCRS FlairBBP

4,9 Bilhões de Tokens

Forward	*2.76*
Backward	*2.80*

FlairBBP - Disponível em:
https://github.com/jneto04/ner-pt

WE Português - Tweeter

BERT Multilingue
(inclui o Português)

BERT - PyTorch

Dump Wikipédia
Top 100

PyTorch - Disponível em:
https://github.com/huggingface/pytorch-pretrained-BERT

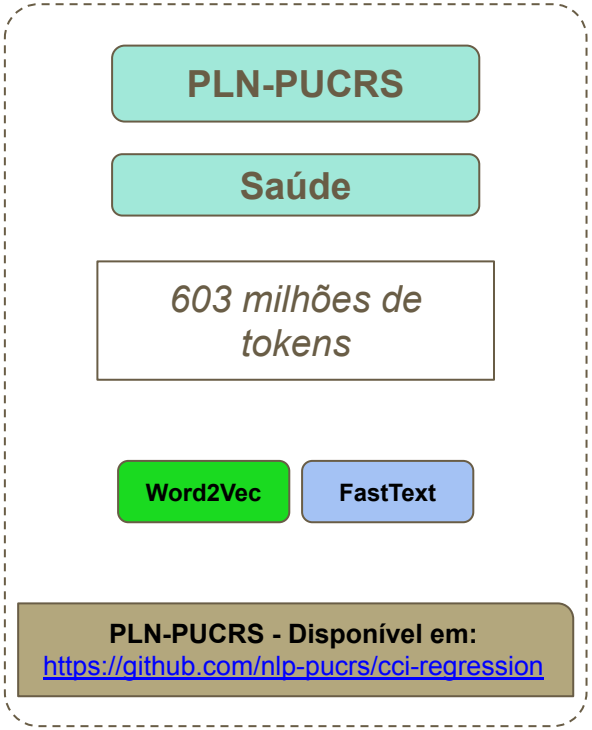
BERT - TensorFlow

Dump Wikipédia
Top 100

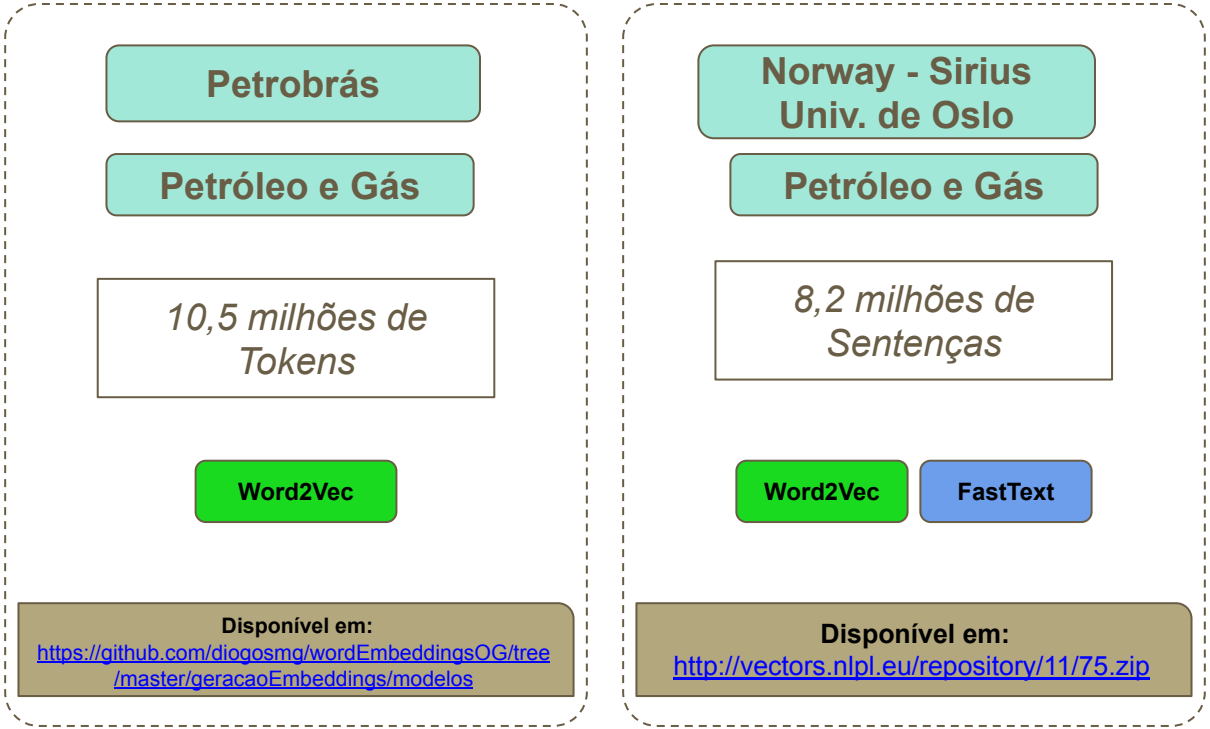
TensorFlow - Disponível em:
https://github.com/google-research/bert

Modelos específicos de domínio

Word Embeddings de domínio



Word Embeddings de domínio



Geração de modelos

TensorFlow

https://www.tensorflow.org

PyTorch

https://pytorch.org/

Gensim

https://radimrehurek.com/gensim/

Aplicações

Aplicações no grupo de PLN PUCRS

Classificação de textos (prontuários eletrônicos)

Recuperação de informação em Geologia

Reconhecimento de Entidades nomeadas:

- Extração de informação em investigação policial
- Extração de informação em inteligência de negócios
- Extração de informação em geologia
- Extração de informação em saúde

Avaliação de modelos

Tipos de avaliação

Intrínseca

Extrínseca

Tipos de avaliação

Modelos WE

- Diferentes idiomas
- Diferentes coberturas:
  - Cobertura geral
  - Cobertura em domínios específicos

Avaliação

Modelos de cobertura geral da língua

Testes de analogia semântica:

- capitais e países comuns
- todas as capitais e países
- países e moedas
- cidades e estados
- gênero em relações familiares (brother, sister) - (irmão, irmã)

Avaliação intrínseca modelo geral

Cobertura geral

Testes de analogía sintática:

- adjetivo ao advérbio (rápido, rapidamente)
- oposto (possível, impossível)
- comparativo (grande, maior)
- superlativo (easy, easiest) - (fácil, o mais fácil)
- particípio presente (pensar, pensante) - gerúndio (pensar, pensando)

Avaliação em domínio específico

Avaliações de WE em domínios específicos, ex. área biomédica

Bruni 2014:

- Dois pares são apresentados a especialistas que devem julgar qual par é mais semelhante entre si.
- Cada par é avaliado com outros 50 pares escolhidos randomicamente, que então recebem um score dado pelo número de vezes que o par foi julgado como o mais similar.
- O modelo WE deve reproduzir a intuição de similaridade dos especialistas.

Avaliação intrínseca em domínio específico

Modelo prévio de domínio Geológico

- Farhad (inglês)
- Petrobrás (português)

Avaliação em domínio específico

Par 1			Par 2		
Cientista	Hidrocarboneto		Montmorilonita	Argilomineral	X
Jurássico	Mesozóico	X	Geofísico	Pacote	

Avaliação

Geração da lista de pares de termos do domínio

Termos:

- Lista gerada por especialistas
- Termos do tesauro

Lista do especialista

Geofísica	Paleontologia	Ambientes Deposicionais	Petrologia Sedimentar	Petrologia Ígnea	Petrologia Metamórfica	Mineralogia
sísmica	plâncton	fluvial	diagênese	biotita	xisto	lâminas
log	diatomáceas	deltaico	sal	quartzo	ardósia	petrografia
sismologia	algas vermelhas	carbonático	carbonato	ortoclásio	gnaisse	minerais
gravimetria	algas verdes	depósito	óleo	lâmina petrográfica	clorita	argila
seção	dinoflagelados	marinho	dolomita	textura	biotita	biotita
perfil	foraminíferos	subambientes	água	mineralogia	silimanita	piroxênio
magnetometria	cianobactérias	plataformas	soterramento	classificação petrográfica	granada	ortopiroxênio
refração	estampa	glacial	porosidade	augita	cianita	quartzo
reflexão	paleocorrente	aluvial	esmectita	hornblenda	feldspato alcalino	plagioclásio
perfilagem		maré	ilita	feldspato alcalino	migmatito	albita
sismoestratigrafia		ondas	arredondado	feldspato potássico		granada
prospecção		costeiro	trama	olivina		zircão
magnetismo		litologia	fábrica	magma		epidoto
seção		sistema	depósito	câmara		arsenopirita
topografia		deposicional	folhelho	magmático		bornita

Tesauro (skos)

<skos:Concept rdf:about="http://bs/#GRAPH">

<skos:prefLabel xml:lang="en">GRAPH</skos:prefLabel>

<skos:prefLabel xml:lang="pt-BR">**Gráfico**</skos:prefLabel>

<skos:altLabel xml:lang="en">CURVE (GRAPH)</skos:altLabel>

<skos:altLabel xml:lang="en">GRAPHICAL CURVE</skos:altLabel>

Tesauro

<skos:broader rdf:resource="http://bs/#CHART"/>

<skos:narrower rdf:resource="http://bs/#AMPLITUDE+CURVE"/>

<skos:related rdf:resource="http://bs/#CURVING"/>

<skos:related rdf:resource="http://bs/#DIAGRAM"/>

Geração dos pares

A partir de termos presentes em tesauro

Termos similares - ''broader" (termos mais amplo), ou "narrower" (termo menos amplo)

Termos levemente similares - "related" (termo relacionado)

Termos menos similares foram estabelecidos como sendo aqueles sem relação

Lista de pares

Combinação de pares

3900 entradas, indicação do mais similar

Par 1			Par 2		
Cientista	Hidrocarboneto		Montmorilonita	Argilomineral	X
Jurássico	Mesozóico	X	Geofísico	Pacote	

(300÷25)×(25×((25+1)÷2)) ou (np÷nr)×(nr×((nr+1)÷2))

Avaliação extrínseca

Aplicação dos modelos em tarefas de outra natureza

Tarefa	WE	ELMo	BERT	Flair
REN	✔ *Lample, 2016*	✔ *Perters, 2018*	✔ *Devlin, 2018*	✔ *Akbik, 2018*
ER	✔ *Lin, 2016*		✔ *Soares, 2019*	
COREF	✔ *Lee, 2017*	✔ *Perters, 2018*	✔ *Joshi, 2019*	
CT	✔ *Joulin, 2016*			✔ *Akbik, 2018*
SS	✔ *Hartmann, 2017*		✔ *ASSIN 2*	

Limitações

- A semântica refletida no modelo incorpora todas as diferentes nuances do significado mas não as diferencia
  - (e.g. casos em que é sinônimo, casos em que é antônimo)
- Custo computacional
- Caixa preta - ideal é IA explicável (explainable AI)
- Semântica na perspectiva da palavra ou termo (discurso, pragmática)
- Limitação sobre o conhecimento da língua por parte de desenvolvedores

Obrigada!

Referências

[Akbik, 2018] Akbik, A., Blythe, D., & Vollgraf, R. (2018). Contextual string embeddings for sequence labeling. In Proceedings of the 27th International Conference on Computational Linguistics (pp. 1638-1649).

[Bojanowski, 2016] Bojanowski, P., Grave, E., Joulin, A., and Mikolov, T. (2016). Enriching Word Vectors with Subword Information. arXiv preprint arXiv:1607.04606.

[Devlin, 2018] Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.

[Levy, 2014] Levy, O.; Goldberg, Y. “Dependency-based word embeddings”. In: Proceedings of the52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), 2014, pp. 302–308.

[Lee, 2017] Lee, K., He, L., Lewis, M., & Zettlemoyer, L. (2017). End-to-end neural coreference resolution. arXiv preprint arXiv:1707.07045.

Referências

[Ling, 2015] Ling, W., Dyer, C., Black, A., and Trancoso, I. (2015). Two/too simple adaptations of word2vec for syntax problems. In Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies.

[Lin, 2016] Lin, Y., Shen, S., Liu, Z., Luan, H., & Sun, M. (2016, August). Neural relation extraction with selective attention over instances. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers) (pp. 2124-2133).

[Lample, 2016] Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K., & Dyer, C. (2016). Neural architectures for named entity recognition. arXiv preprint arXiv:1603.01360.

[Mikolov, 2013] Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G. S.; Dean, J. “Distributed representations of words and phrases and their compositionality”. In: Advances in neural information processing systems, 2013, pp. 3111–3119.

[Mikolov and Zweig, 2012] T. Mikolov and G. Zweig. Context dependent recurrent neural network language model. In IEEE SLT, pages 234–239, 2012.

Referências

[Mikolov et al., 2010] T. Mikolov, M. Karafiat, L. Burget, J. Cernocky, and S. Khudanpur. Recurrent neural network based language model. In INTERSPEECH, pages 1045–1048, 2010.

[Mikolov et al., 2011a] T. Mikolov, S. Kombrink, L. Burget, J. Cernocky, and S. Khudanpur. Extensions of recurrent neural network language model. In Proc. of IEEE ICASSP, pages 5528–5531, 2011.

[Mikolov et al., 2012] T. Mikolov, I. Sutskever, A. Deoras, H. Le, and S. Kombrink. Subword Language Modeling With Neural Networks. 2012.

[Mikolov et al., 2013] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. CoRR, abs/1301.3781, 2013.

[Pennington, 2014] Pennington, J., Socher, R., & Manning, C. (2014). Glove: Global vectors for word representation. In Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP) (pp. 1532-1543).

Referências

[Perters, 2018] Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., & Zettlemoyer, L. (2018). Deep contextualized word representations. arXiv preprint arXiv:1802.05365.

[Rosenfeld, 200] Rosenfeld, R. (2000). Two decades of statistical language modeling: Where do we go from here?. Proceedings of the IEEE, 88(8), 1270-1278.

[Soares, 2019] Soares, Livio Baldini, et al. "Matching the Blanks: Distributional Similarity for Relation Learning." arXiv preprint arXiv:1906.03158 (2019).

[Jozefowicz, 2015] Rafal Jozefowicz, Oriol Vinyals, Mike Schuster, Noam Shazeer, and Yonghui Wu. 2016. Exploring the limits of language modeling. CoRR abs/1602.02410.

[Joulin, 2016] Joulin, A., Grave, E., Bojanowski, P., & Mikolov, T. (2016). Bag of tricks for efficient text classification. arXiv preprint arXiv:1607.01759.

Referências

[Joshi, 2019] Joshi, M., Levy, O., Weld, D. S., & Zettlemoyer, L. (2019). BERT for Coreference Resolution: Baselines and Analysis. arXiv preprint arXiv:1908.09091.

[Zhang, 2015] Zhang, D.; Xu, H.; Su, Z.; Xu, Y. “Chinese comments sentiment classification based on word2vec and svm perf”,Expert Systems with Applications, vol. 42–4, 2015, pp.1857–1863.