Deep Learning for Natural Language Processing (in 2+ hours)

Eneko Agirre
http://ixa2.si.ehu.eus/eneko
@eagirre

IXA taldea http://ixa.eus
HiTZ Research center https://hitz.eus

Contents

- Introduction to NLP
  - Deep Learning ~ Learning Representations
- Text as bag of words
  - Text Classification
  - Representation learning and word embeddings
  - Superhuman: xlingual word embeddings https://www.aclweb.org/anthology/P18-1073/
- Text as a sequence RNN (LSTM), attention, transformers
  - Seq2seq, Machine Translation
  - Superhuman: unsupervised MT https://www.aclweb.org/anthology/P19-1019/

Natural Language Processing

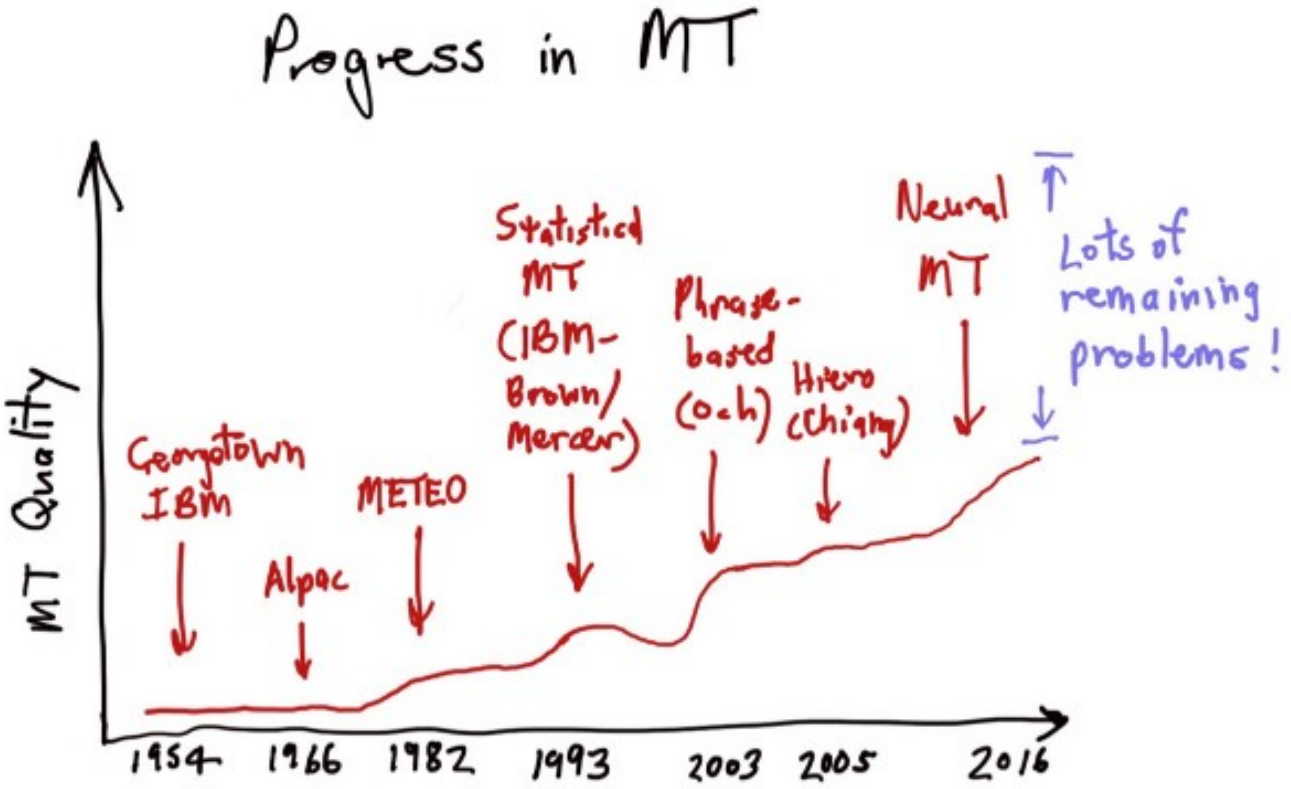
- Intersection of AI and linguistics
  - Machine learning
  - Big data
  - Data science

  aka text processing, text analytics,
  natural language engineering,
  computational linguistics, …
- Goal: process natural language input
  to perform a task
- Language understanding is AI-complete

  Youth unemployment around 37%, wow!

NLP tasks – Machine Translation

Given sentence or sets of sentences output translation in other languages

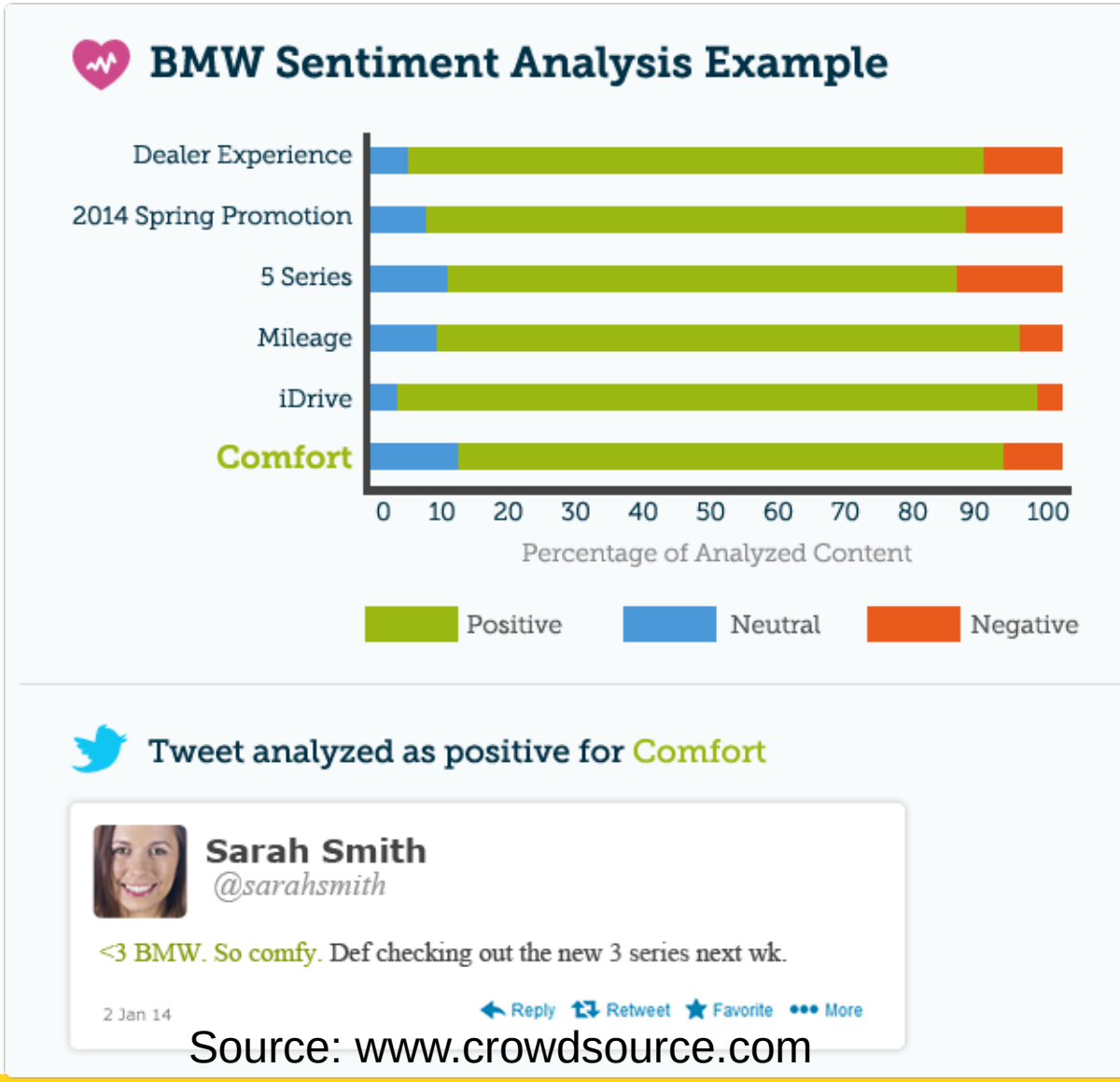


Source: Chris Manning

NLP tasks – Sentiment Analysis

Given sentence
or short document

text is positive/
negative/neutral



Source: www.crowdsource.com

NLP tasks – Question Answering

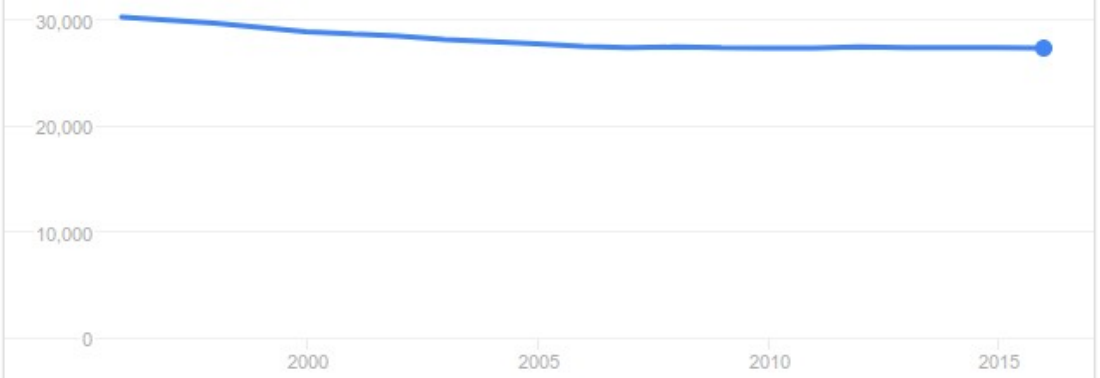
how many people live in eibar?

All Maps News Images Shopping More Settings Tools

About 1,570,000 results (0.72 seconds)

Eibar / Population

27,380 (2016)



Explore more

Sources include: Instituto Nacional de Estadística Feedback

Eibar

City in Spain

Eibar is a city and municipality within the province of Gipuzkoa, in the Basque Country of Spain. It is the head town of Debabarrena, one of the comarcas of Gipuzkoa. Eibar has 27,439 inhabitants. Wikipedia

Elevation: 121 m

Area: 24.78 km²

Weather: 7°C, Wind S at 18 km/h, 81% Humidity

Population: 27,380 (2016) Instituto Nacional de Estadística

Hotels: 3-star averaging €67. View hotels

Source: google.com

NLP tasks – (Spoken) dialogue

Simple dialogues with Alexa, OK google, Siri:

What's in the news?
What's the weather like?
What's my commute look like?
Add eggs to my shopping list
I need to buy laundry detergent.

Source: www.cnet.com/how-to/
the-complete-list-of-siri-commands/
complete-list-of-ok-google-commands/
amazon-echo-the-complete-list-of-alexa-commands/

Alexa Prize: socialbots

User: Let's chat about the Mars Mission.
Socialbot: There are multiple Mars missions, some public and some private.
User: Who do you think will succeed?
Socialbot: I think more than one will succeed. The first one to get to Mars will open the doors for others.
User: I'd love to go to Mars.
Socialbot: Me too. Luckily I can hitch a ride on any computer.
User: That's very funny.
Socialbot: The biggest challenge isn't technology, it's funding. It will cost $500 billion to send humans to Mars.


amazon


Maria wikizalea
Kaixo! Ni Maria naiz, wikizalea. Esadazu lehenengo zeri buruz jakin nahi duzun zerbait, eta, ondoren, zure galderak erantzuten saiatuko naiz.
Hitz zentroa interesatzen zait

NLP tasks – getting mainstream

- Search engines
- Machine (assisted) translation
- Spelling, grammar checking
- Online advertisement placement
- Sentiment analysis for finance, reputation
- Voice commands
- Voice assistants

NLP levels

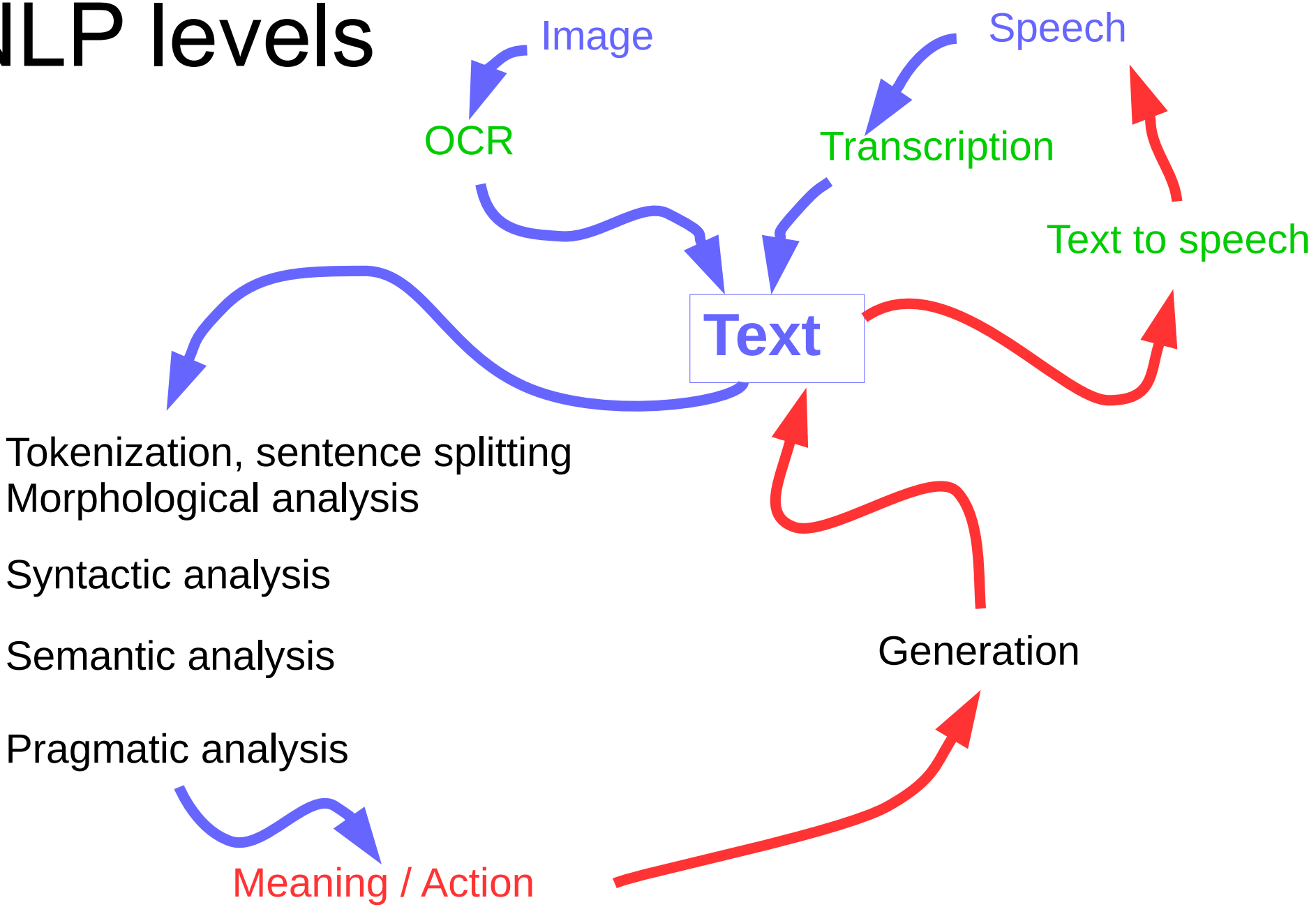


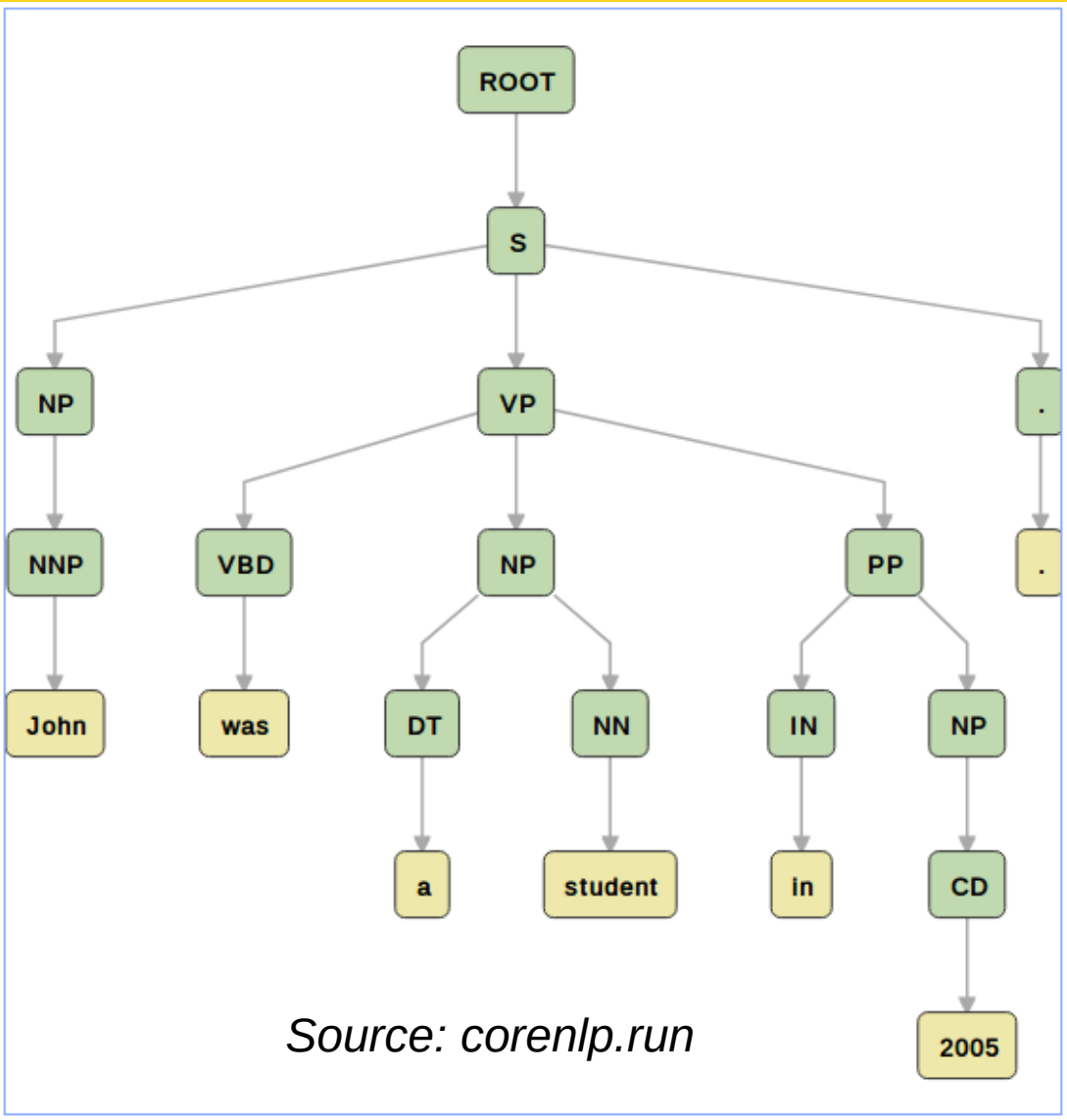
Image
Speech
OCR
Transcription
Text to speech
Text
Tokenization, sentence splitting
Morphological analysis
Syntactic analysis
Semantic analysis
Pragmatic analysis
Generation
Meaning / Action

NLP levels

- Tokenization, sentence splitting
  - *John was a student in 2005.*
  - *2005. urtean John ikaslea zen.*
  - 约翰是 2005 年的学生。
- Morphological analysis (PoS, lemma, NERC)
  - *was* (lemma *be,* PoS *V)*
  - *John F. Kennedy* (lemma *John_F._Kennedy*, PoS *NNP*, NERC *Person*)
  - *etxekoarentzat* (lemma *etxe*, PoS N, for the one of the house)
  - *tomemos* (lemma *tomar*, PoS V, let we take it)

NLP levels

- Syntactic analysis



Source: corenlp.run

- Semantic analysis

$$\exists x, y \wedge name(x, \text{John}) \wedge student(x)$$
$$intime(x, y) \wedge time(y, \text{T2005xxxx})$$

Source: gmb.let.rug.nl

NLP levels

- Pragmatics (discourse, correference, ...)
  - ***But** Mary become a lawyer **that year**.*
  - Wasn't it one year later?
- Inference
  - *John and Mary were law students in Dec. 2005*
  - *Mary was working full-time as a lawyer in 2005*

NLP is difficult

- Ambiguity at all levels
  - *cells in prisons* vs. *cells in animals*
  - *One morning I shot an elephant in my pajamas.*
    How he got into my pajamas I'll never know. (Groucho)
  - *You mean Mary Smith or Mary Doe?*
- Variability at all levels (many ways to convey a meaning)
- Subtlety
- Understanding language requires
  - Language knowledge (word meaning, grammar, ...)
  - World knowledge (physical, encyclopedic, visual ...)
  - Common sense and inference ability
- But sometimes it is surprisingly easy!

Brief history of NLP

- 1960s: Complex rules and first order logic.
  Humans build complex grammars.
- 1990s: Supervised machine learning.
  Humans annotate text,
  design laborious task-specific features,
  and apply ML techniques.
- 2010s: Deep learning.
  Learning continuous representations,
  get rid of task-specific features.

Quiz

What is deep learning?

Why deep learning for NLP?

- Technology behind current speech processing and machine translation
  - Advances in the state-of-the-art of most tasks
- Focus on representation learning
  - Learns a representation for words and word sequences that is fitted to the task
    … including world and visual knowledge.
  - Naturally accounts for graded judgements about language:
    - Word similarity: building / house
    - Sentence similarity:
      A pony is close to the house / There is a horse in the front yard
- End-to-end joint learning (vs. pipeline)
- Transfer models between tasks (word embeddings, pre-trained LM)
- But deep learning has its limitations too!!

Contents

- Introduction to NLP
  - Deep Learning ~ Learning Representations
- Text as bag of words
  - Text Classification
  - Representation learning and word embeddings
  - Superhuman: xlingual word embeddings https://www.aclweb.org/anthology/P18-1073/
- Text as a sequence RNN (LSTM), attention, transformers
  - Seq2seq, Machine Translation
  - Superhuman: unsupervised MT https://www.aclweb.org/anthology/P19-1019/

A sample NLP task with ML

Text classification

- Spam or not
- Positive or negative movie review
  - Full of zany characters and richly applied satire
  - It was pathetic
  - Modestly accomplished, lifted by two terrific performances.
  - Not NEARLY as funny as its title

A sample NLP task with ML

Text classification:

- Input
  - A document d $\in$ D
  - A fixed set of classes C={c_1, c_2, … c_j}
- Output: a predicted class y $\in$ C

Text classification as regression:

- Input
  - A document d $\in$ D
  - A range of real values C=[0,j]
- Output: a predicted value y $\in$ C

A sample NLP task with ML

Text classification method 1: Hand-coded rules

- Rules based on combination of words and other features
  - emoji ∈ { 🙂 😂 } => positive
  - word ∈ { terrible, ugly } => negative
- Accuracy very high
  - If rules refined by expert
- But writing and maintenance very expensive

A sample NLP task with ML

Text classification method 2: Supervised ML

Learn a classifier from hand-annotated examples

- Input: A training set of n hand-labeled documents $(x_1,y_1) \ldots (x_m, y_m)$
- Output: A learned classifier $f:D \rightarrow C$
- Recall very high
- Cost-effective: experts only needed for annotation

Supervised doc. classification

Representation of each example (document)

- Key idea for most machine learning
- Example as a vector of features x
  - All examples same number of features
  - Features: boolean, integer, real
- Pre-processing code
  to convert from example into feature vector
  - Substantial effort

Supervised doc. classification

Representation of each example (document)

⇒ Bag of words representation

I love this movie! It's sweet, but with satirical humor. The dialogue is great and the adventure scenes are fun... It manages to be whimsical and romantic while laughing at the conventions of the fairy tale genre. I would recommend it to just about anyone. I've seen it several times, and I'm always happy to see it again whenever I have a friend who hasn't seen it yet!

it	6
I	5
the	4
to	3
and	3
seen	2
yet	1
would	1
whimsical	1
times	1
sweet	1
satirical	1
adventure	1
genre	1
fairy	1
humor	1
have	1
great	1
…	…

Source: Sam Bowman

Supervised doc. classification

Representation of each example (document)

⇒ Bag of words representation

Limited number of words (size of vocabulary)!
Each word is a feature:
non-negative integer or boolean:

it	6
I	5
the	4
to	4
and	3
...	

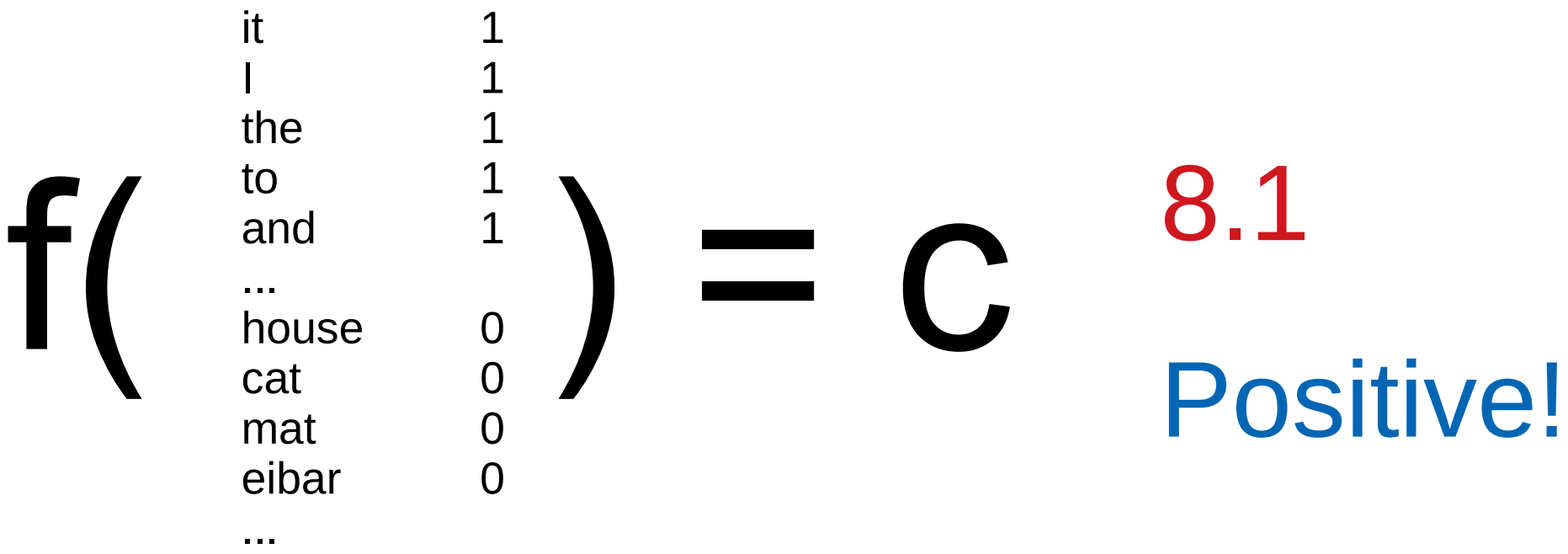
it	1	house	0
I	1	cat	0
the	1	mat	0
to	1	him	0
and	1	eibar	0
...		...	

Supervised doc. classification

Output of the classifier: continuous or discrete

Sentiment analysis:

real number [0:10] or two classes



Supervised doc. classification: continuous

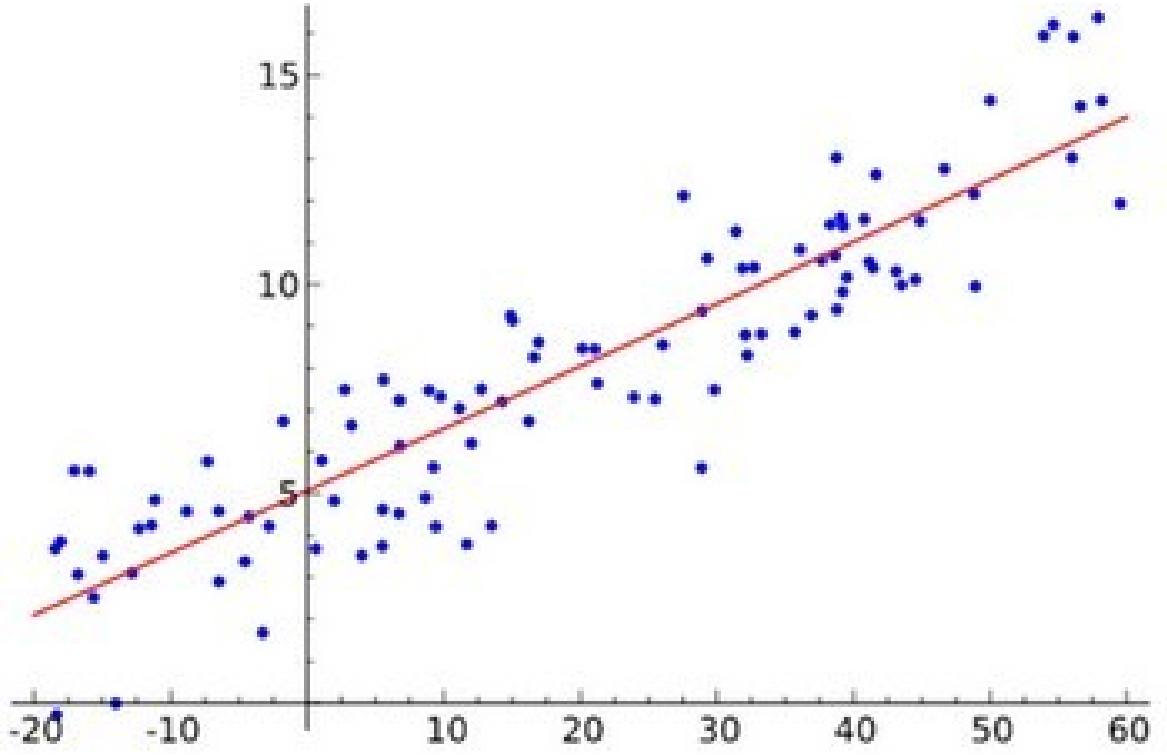
Linear regression (continuous class)

- Assign a weight to each feature: vector of weights w
- Output value dot product: $y = w^T x + b$
  - e.g. with three features: $y = w_1x_1 + w_2x_2 + w_3x_3 + b$
- Task for linear regression:
  - Given labeled examples
  - Find w such that the output value (y) is not too wrong for as many training examples as possible

Supervised doc. classification: continuous

Linear regression

- with one feature: $y = w_1 x_1 + b$



Source: gerardnico.com/wiki/data_mining/linear_regression

Supervised doc. Classification: discrete

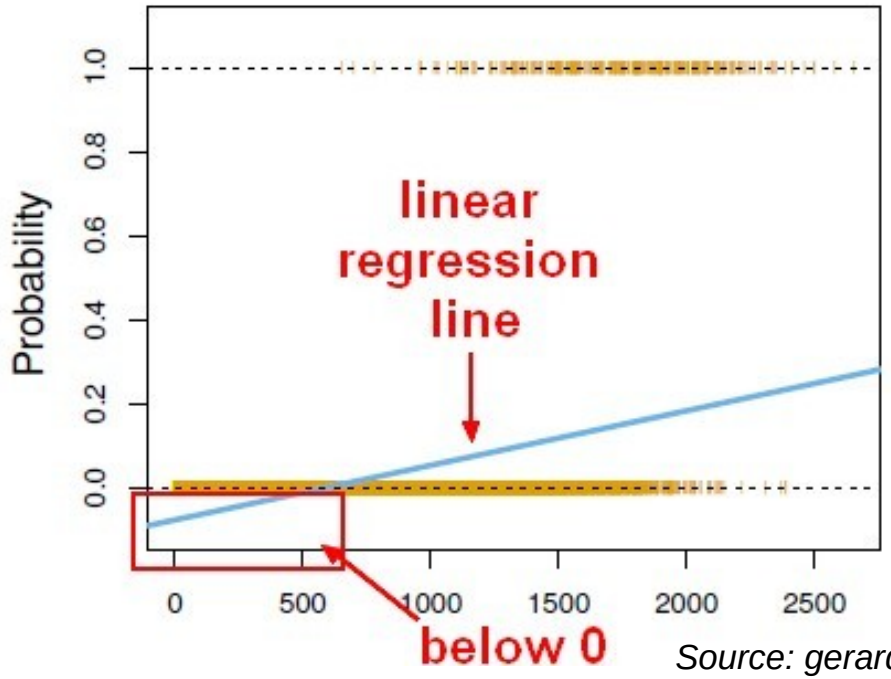
Linear classification: (Multinomial) logistic regression

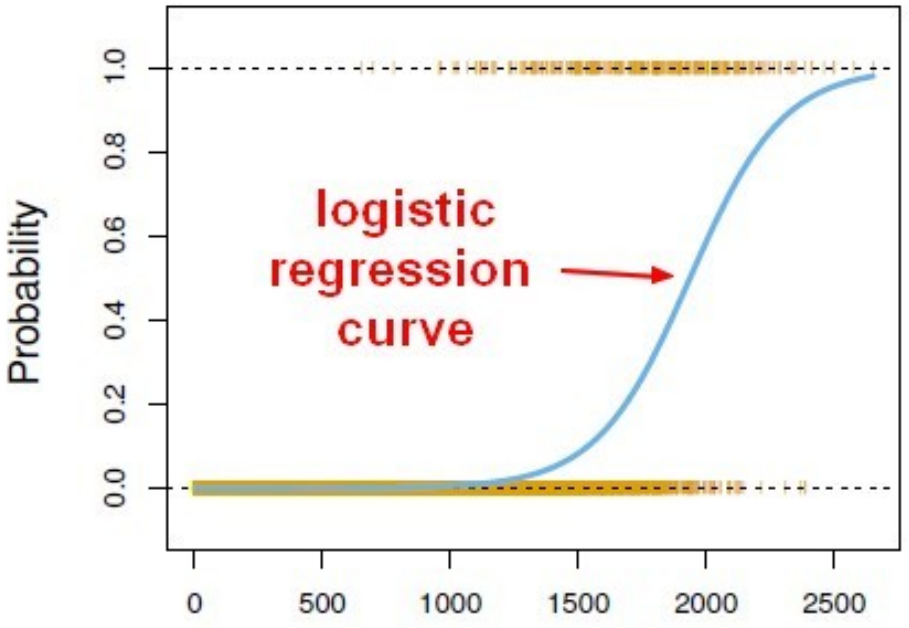
- Assign a weight to each feature for each class: vectors of weights w_c for class c
  - For each class compute $w_c^T x$
  - Add non-linearity $f_c = \exp(w_c^T x)$
  - Normalize it to estimate probabilities: $p(y=c|x) \sim f_c / \sum_{c' \in C} f_{c'}$
- Output value $y = \text{argmax}_c \; p(c|x)$
- Task for linear classification:
  - Given labeled examples
  - Find w_c vectors such that the output value is not wrong for as many training examples as possible

Supervised doc. classification: discrete

(multinomial) logistic regression
= softmax classification
≠ Naive Bayes, Support Vector Machine

Why $\exp(w_c^T x)$?





Source: gerardnico.com/wiki/data_mining/simple_logistic_regression

Supervised doc. classification
Softmax classification

Estimating parameters

- Choose parameter which minimize error over training data

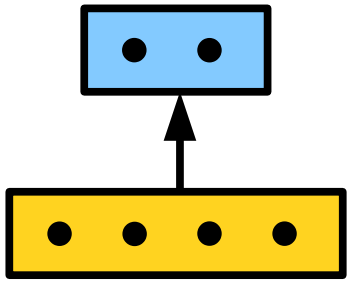
  Loss function J (aka cost f. or objective f.)

  Cross-entropy error on one example (x_i,y_i)

  - We want to maximize probability of correct class
    i.e. **minimize** negative log probability of the correct class

$$J_i(W) = -\log P(y_i = c | x_i) = -\log \left(\frac{\exp(W_c x)}{\sum_{c' \in C} \exp(W_{c'} x)} \right)$$

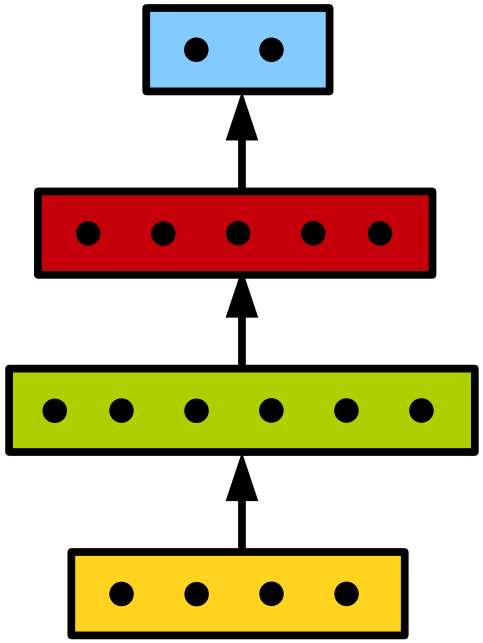
Deep: Multilayer perceptron



Logistic Regression

- An input layer – just a feature vector
- An output layer – class probabilities

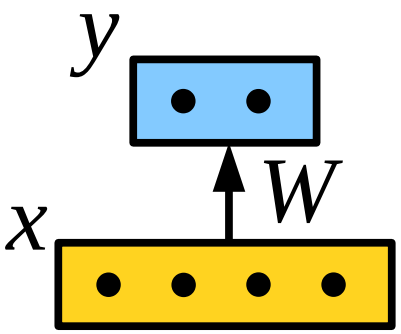
Deep: Multilayer perceptron



- An input layer
  – just a feature vector
- One or more hidden layers, each computed on the layer below
  – latent features.
- An output layer, based on the top hidden layer
  – class probabilities
- Also known as Feed Forward

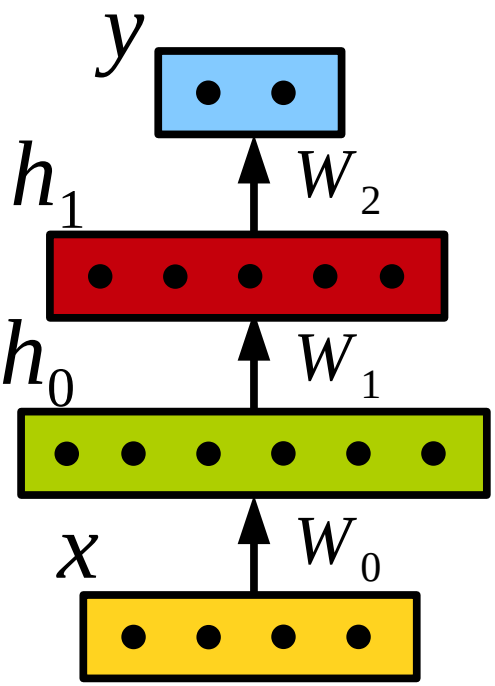
Deep: Multilayer perceptron

Logistic Regression



$$y = softmax(xW + b)$$

Deep: Multilayer perceptron

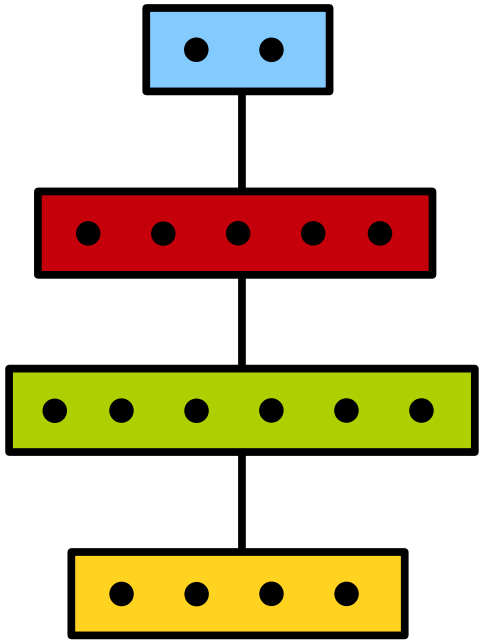


$$y = softmax(h_1 W_2 + b_2)$$

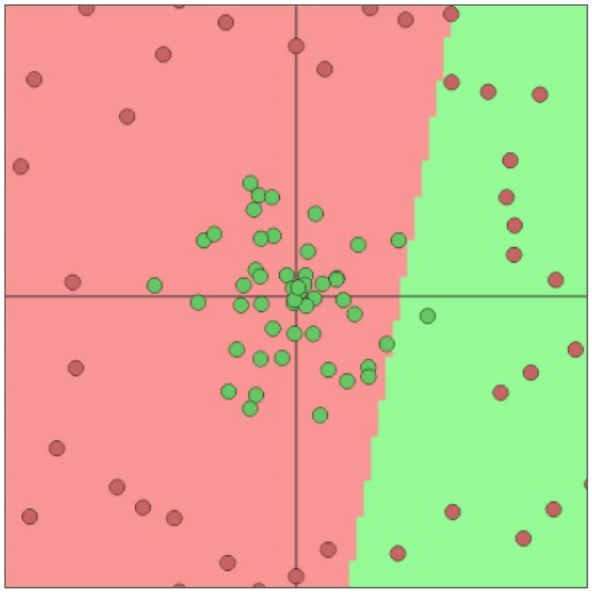
$$h_1 = f(h_0 W_1 + b_1)$$

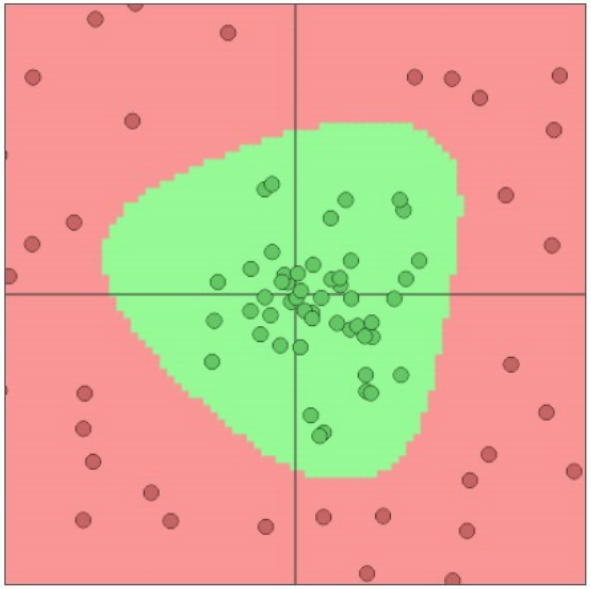
$$h_0 = f(x W_0 + b_0)$$

Deep: Multilayer perceptron



Motivations?





Source: http://cs.stanford.edu/people/karpathy/convnetjs/demo/classify2d.html

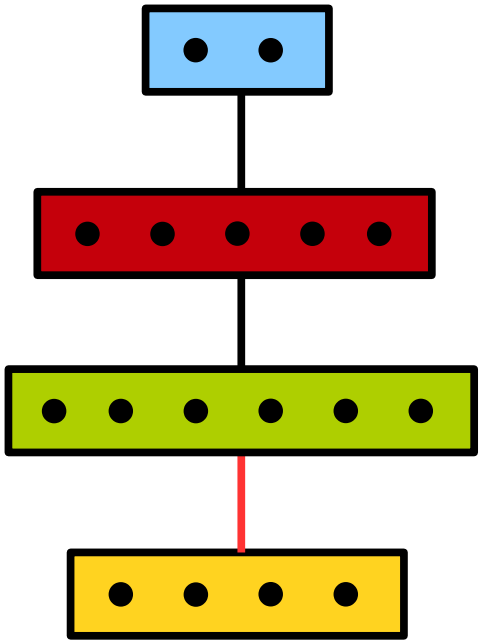
Without non-linearities, there is no extra expresivity: a sequence of linear transformations is a linear transformation

Representation learning

- Document as **bag-of-words**:
  - Each word ~ a **one-hot-vector**
    (0 0 0 0 0 … 1 … 0 0 0)
  - Each document ~ **summation of words in the document**
    (0 0 0 1 0 … 1 … 1 0 0)
- NLP has been representing **words as distinct features** for many years!
  - Is there a better alternative?

Source: Sam Bowman

Representation learning



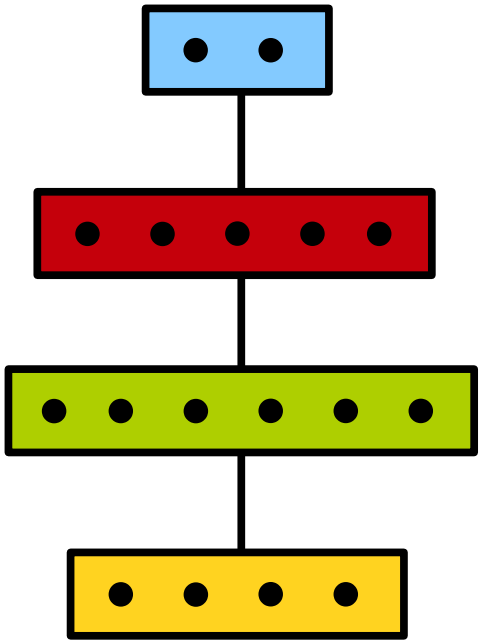
MLP: What are we learning in W_0 when we backpropagate?

$$y = softmax(W_2 h_1 + b_2)$$

$$h_1 = f(W_1 h_0 + b_1)$$

$$h_0 = f(W_0 x + b_0)$$

Representation learning



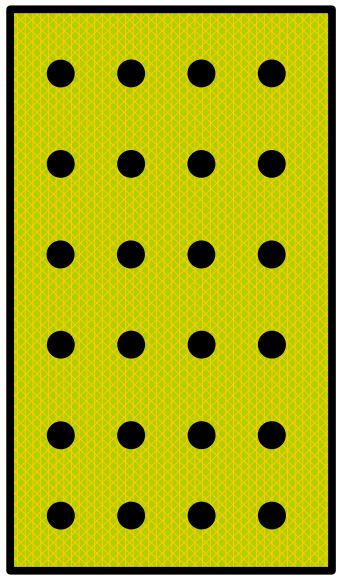
MLP: What are we learning in W_0 when we backpropagate?

$$y = softmax(W_2 h_1 + b_2)$$

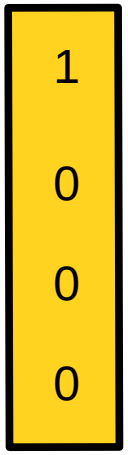
$$h_1 = f(W_1 h_0 + b_1)$$

$$h_0 = f(W_0 x + b_0)$$

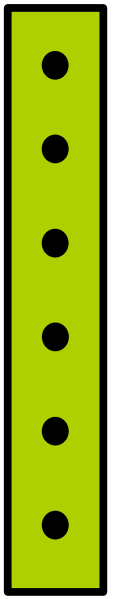
h_0



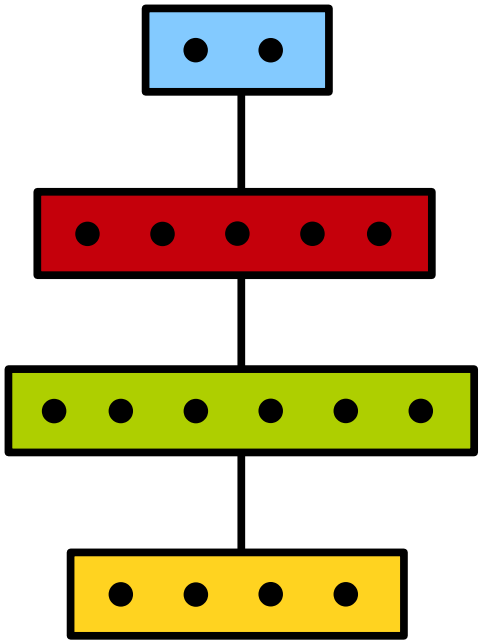
x
(one-hot)



$\underline{h_0}$



Representation learning

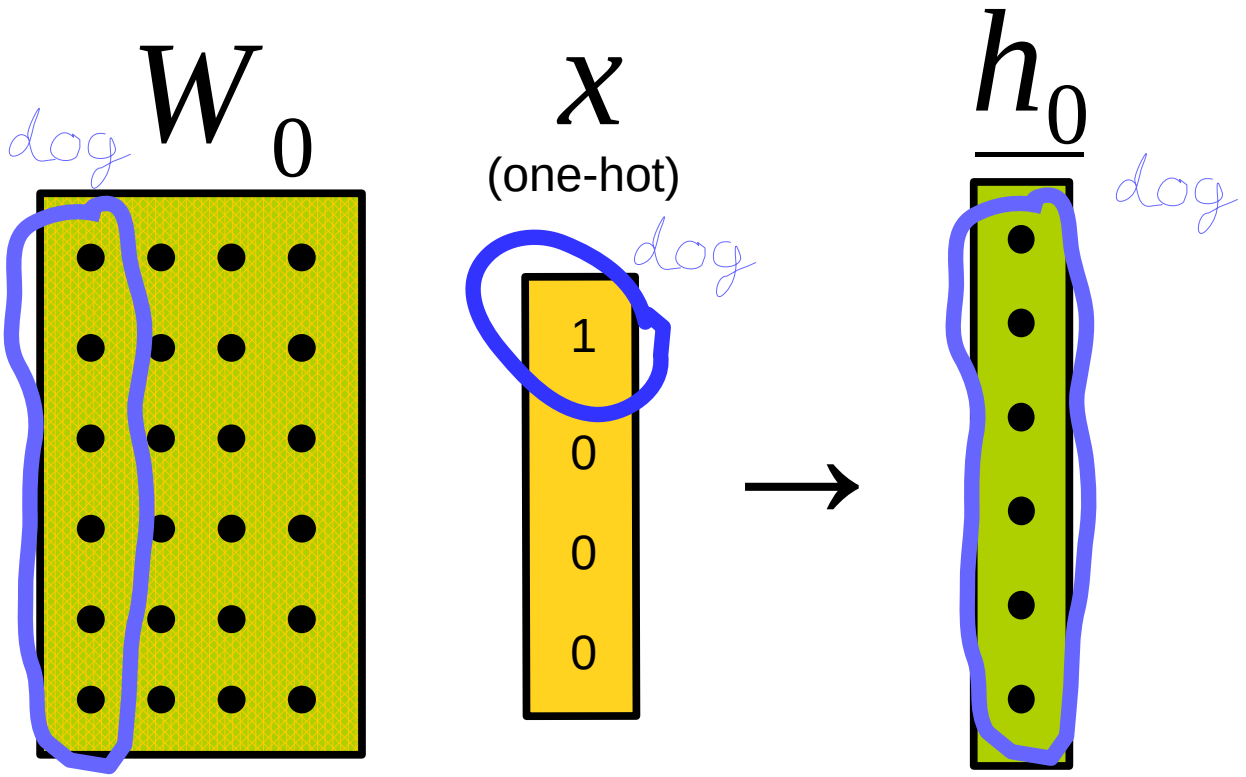


Back-propagation allows Neural Networks to **learn representations for words** while training: **word embeddings!**

$$y = softmax(W_2 h_1 + b_2)$$

$$h_1 = f(W_1 h_0 + b_1)$$

$$h_0 = f(W_0 x + b_0)$$



Representation learning

- Word embeddings
  aka continuous vector space,
  distributed representations
  - Are they useful for anything?
  - Can we transfer this representation
    from one task to the other?
  - Which task would be the best to learn embeddings
    that can be used in other tasks?
  - Can we have all languages in one embedding space?

Contents

- Introduction to NLP
  - Deep Learning ~ Learning Representations
- Text as bag of words
  - Text Classification
  - Representation learning and word embeddings
  - Superhuman: xlingual word embeddings https://www.aclweb.org/anthology/P18-1073/
- Text as a sequence RNN (LSTM), attention, transformers
  - Seq2seq, Machine Translation
  - Superhuman: unsupervised MT https://www.aclweb.org/anthology/P19-1019/

Word embeddings

- Let's represent words as vectors: Similar words should have vectors which are close to each other

WHY?

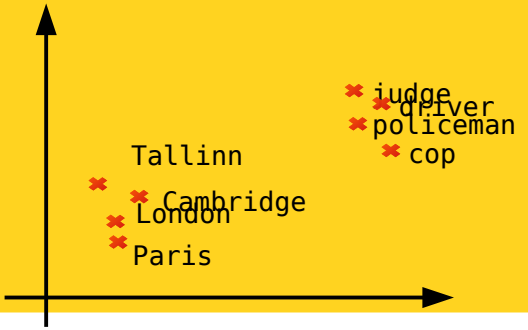
- If an AI has seen these two sequences

```
I live in Cambridge
I live in Paris
```

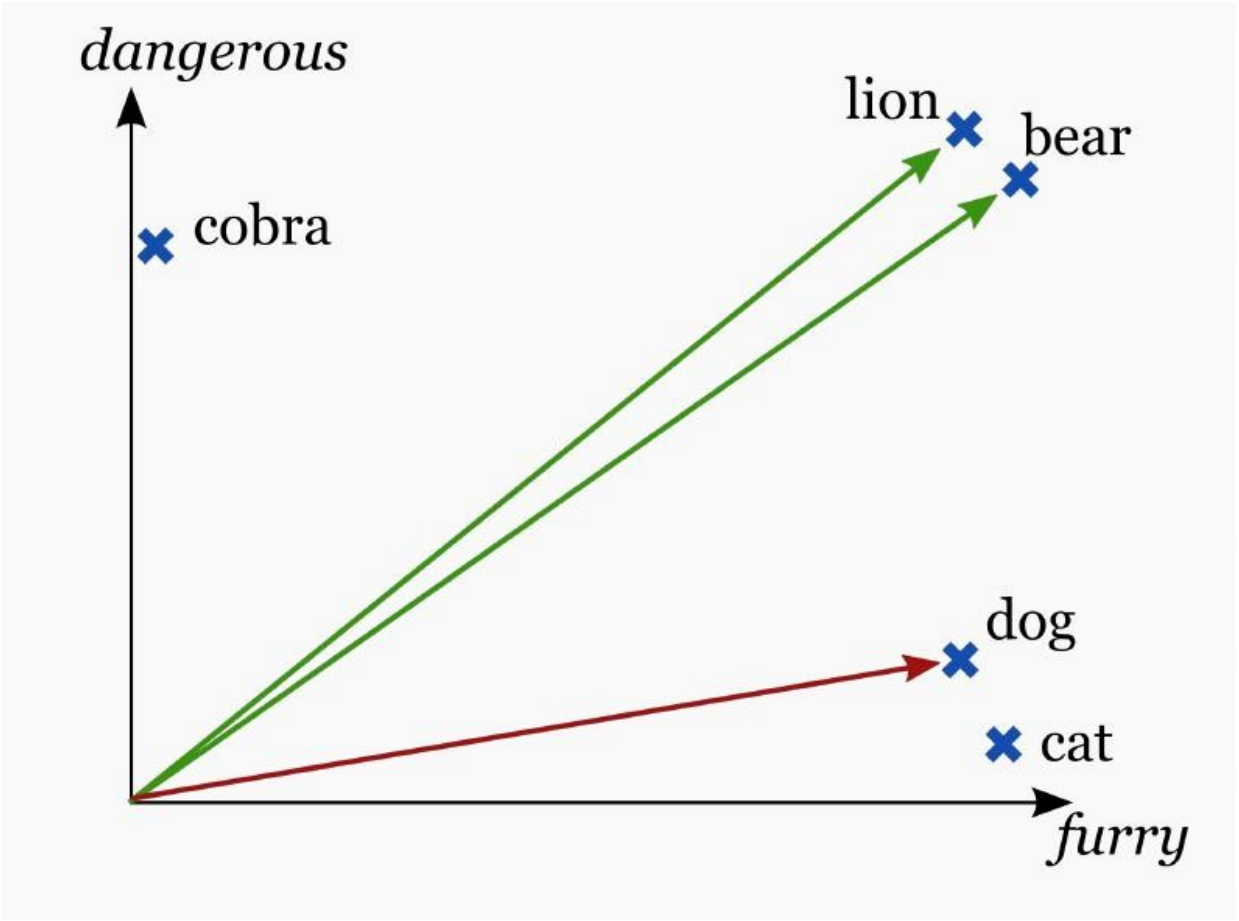
- … then which one should be more plausible?

```
I live in Tallinn
I live in policeman
```

Word embeddings



- Word similarity using cosine



$$\cos(\vec{a},\vec{b}) = \frac{\vec{a}\,\vec{b}^{T}}{\|\vec{a}\|\|\vec{b}\|}$$

$$\cos(\vec{a},\vec{b}) = \frac{\sum_i a_i b_i}{\sqrt{\sum_i a_i^2}\sqrt{\sum_i b_i^2}}$$

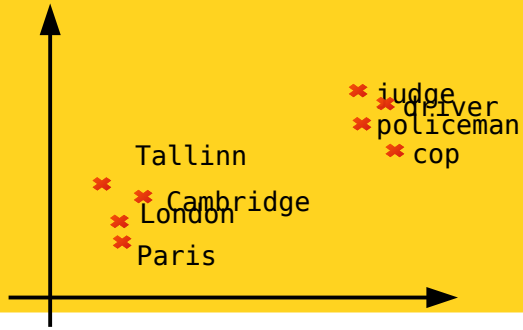
$$\cos(lion, bear) = 0.998$$

$$\cos(lion, dog) = 0.809$$

$$\cos(cobra, dog) = 0.727$$

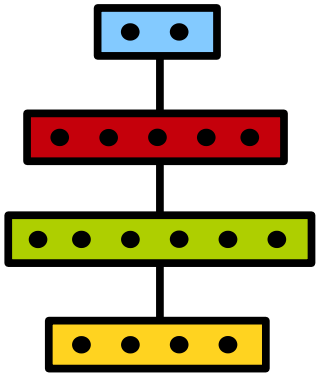
source:http://www.marekrei.com/teaching/cewe/

Word embeddings



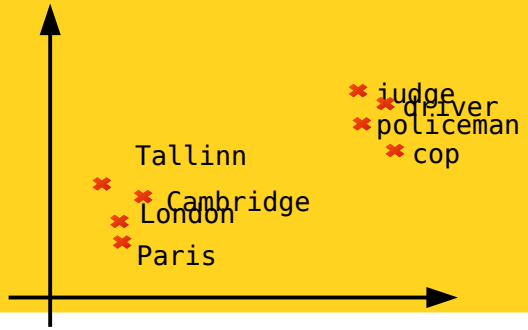
How do we learn vectors for words?

1) Train a MLP on a particular classification task and extract word embeddings



2) Which classification task?

Word embeddings



General task with large quantities of data: **guess the missing word** (language models)

CBOW: given context guess middle word

> *… people who keep pet dogs or **cats** exhibit better mental and physical health …*

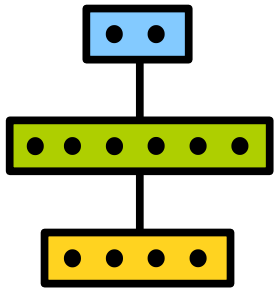
SKIP-GRAM given middle word guess context

> *… people who keep pet dogs or **cats** exhibit better mental and physical health …*

Proposed by Mikolov et al. (2013)

Skip-gram

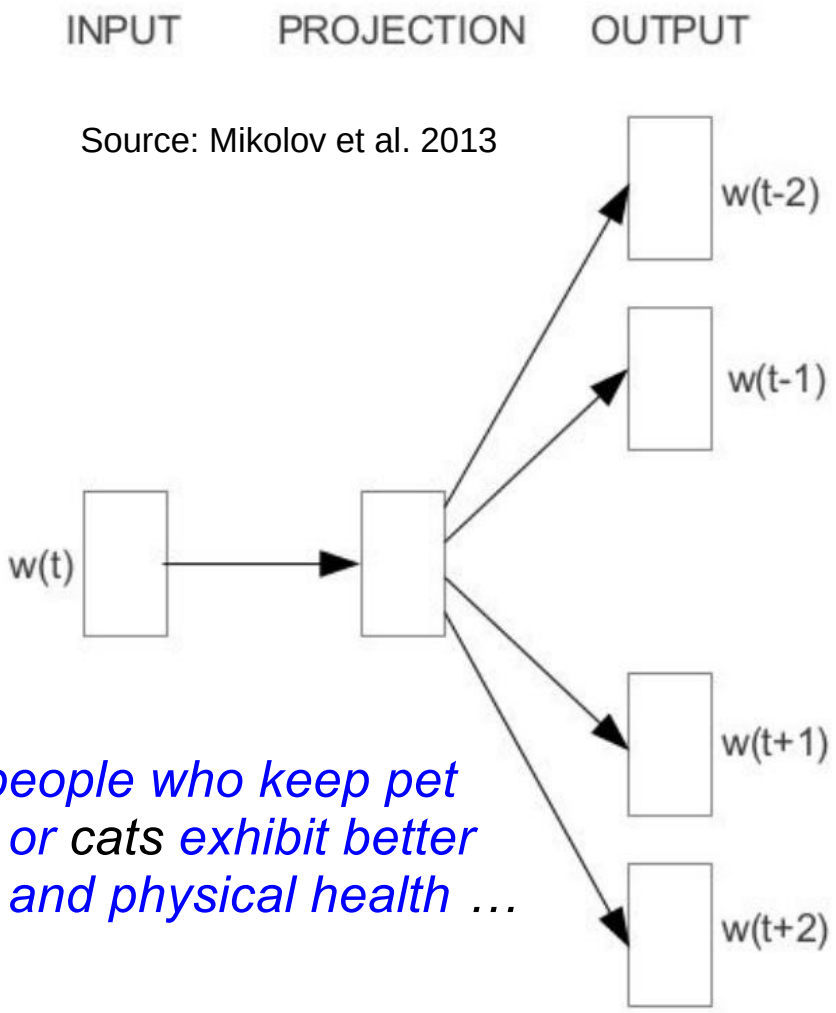
Like MLP, one layer, LARGE **vocabulary**



$$h_0 = W_0 x_t$$

$$y = softmax(W_1 h_0 + b_1)$$

$$y_i = \exp(\underline{h_{0i}}) / \sum_j^V \exp(\underline{h_{oj}})$$



Source: Mikolov et al. 2013

… people who keep pet dogs or cats exhibit better mental and physical health …

Losses:

$$J_{w(t)} = \frac{1}{n} \sum_{i=1}^{n} [\underline{h_{0i}} - \log \sum_j^V \exp(\underline{h_{oj}})]$$

$$J_{\text{NEG}w(t)} = \frac{1}{n} \sum_{i=1}^{n} [\log \sigma(\underline{h_{0i}}) + \sum_{k=1}^{K} E_{w_n \sim P_{noise}} \log \sigma(-\underline{h_{on}})]$$

Word embeddings

Quality of word embeddings evaluated in:

- Word similarity
- Word analogy
- Other tasks like PoS tagging, NERC, sentiment analysis, etc.

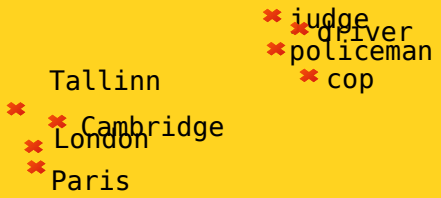
Word embeddings

Word similarity

FRANCE	JESUS	XBOX	REDDISH	SCRATCHED	MEGABITS
454	1973	6909	11724	29869	87025
AUSTRIA	GOD	AMIGA	GREENISH	NAILED	OCTETS
BELGIUM	SATI	PLAYSTATION	BLUISH	SMASHED	MB/S
GERMANY	CHRIST	MSX	PINKISH	PUNCHED	BIT/S
ITALY	SATAN	IPOD	PURPLISH	POPPED	BAUD
GREECE	KALI	SEGA	BROWNISH	CRIMPED	CARATS
SWEDEN	INDRA	PSNUMBER	GREYISH	SCRAPED	KBIT/S
NORWAY	VISHNU	HD	GRAYISH	SCREWED	MEGAHERTZ
EUROPE	ANANDA	DREAMCAST	WHITISH	SECTIONED	MEGAPIXELS
HUNGARY	PARVATI	GEFORCE	SILVERY	SLASHED	GBIT/S
SWITZERLAND	GRACE	CAPCOM	YELLOWISH	RIPPED	AMPERES

Source: Collobert et al. 2011

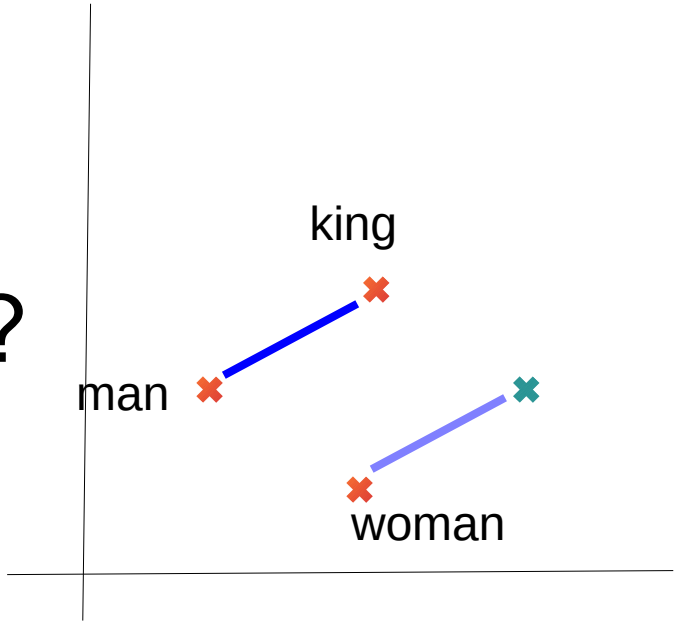
Word embeddings



Word analogy:

a is to b as c is to ?

man is to king as woman is to ?

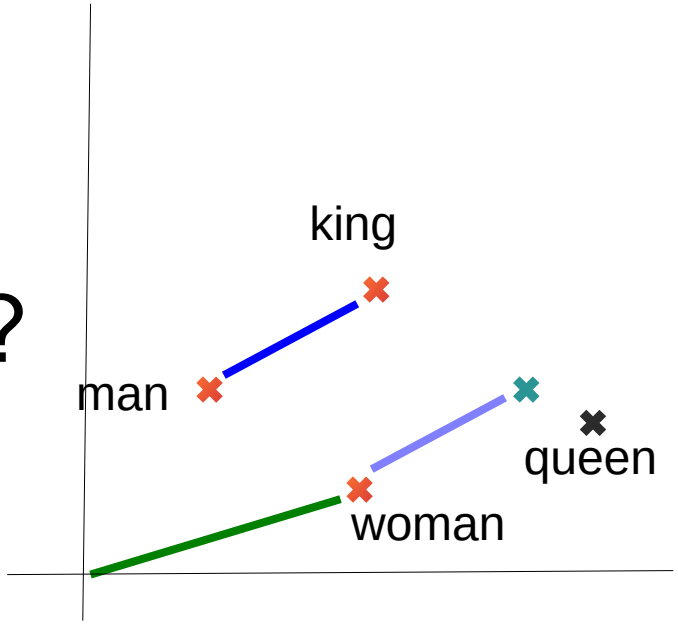


Word embeddings

Word analogy:

a is to b as c is to ?

man is to king as woman is to ?



$$a - b \approx c - d$$

$$d \approx c - a + b$$

$$argmax_{d \in V}\left(\cos\left(d, c - a + b\right)\right)$$

$$king - man + woman \approx queen$$

Word embeddings

Word analogy

Relationship	Example 1	Example 2	Example 3
France - Paris	Italy: Rome	Japan: Tokyo	Florida: Tallahassee
big - bigger	small: larger	cold: colder	quick: quicker
Miami - Florida	Baltimore: Maryland	Dallas: Texas	Kona: Hawaii
Einstein - scientist	Messi: midfielder	Mozart: violinist	Picasso: painter
Sarkozy - France	Berlusconi: Italy	Merkel: Germany	Koizumi: Japan
copper - Cu	zinc: Zn	gold: Au	uranium: plutonium
Berlusconi - Silvio	Sarkozy: Nicolas	Putin: Medvedev	Obama: Barack
Microsoft - Windows	Google: Android	IBM: Linux	Apple: iPhone
Microsoft - Ballmer	Google: Yahoo	IBM: McNealy	Apple: Jobs
Japan - sushi	Germany: bratwurst	France: tapas	USA: pizza

Recap

- Are they useful for anything?
- Can we transfer this representation from one task to the other?
- Which task would be the best to learn embeddings that can be used in other tasks?
- Can we have all languages in one embedding space?

Cross-linguality and machine translation without bilingual data

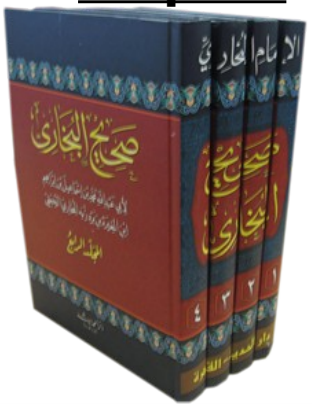
Eneko Agirre
@eagirre

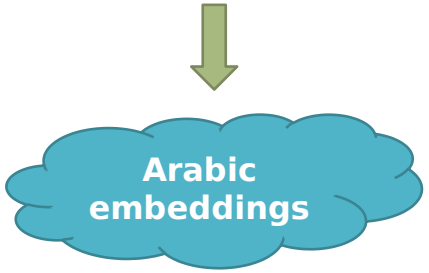
Joint work with: Mikel Artetxe, Gorka Labaka


HiTZ NLP research center - http://hitz.eus
University of the Basque Country (UPV/EHU)

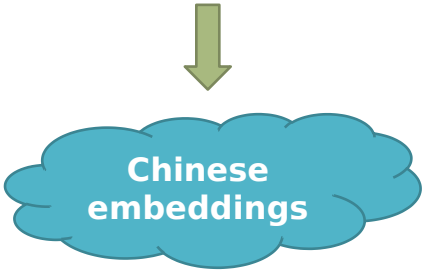
Overview

Arabic monolingual corpora



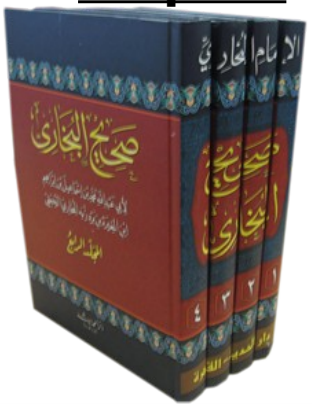


Chinese monolingual corpora



Overview

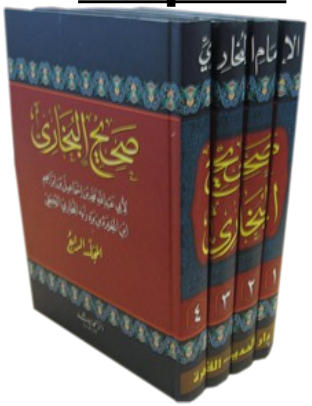
Arabic monolingual corpora



Chinese monolingual corpora

Overview

Arabic monolingual corpora



Chinese monolingual corpora

Arabic embeddings

Chinese embeddings

Bilingual embeddings

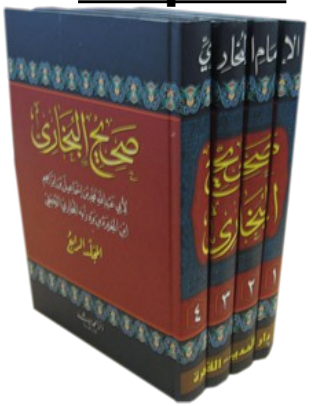
Bilingual dictionaries

Crosslingual & multilingual applications

Machine translation

Overview

Arabic monolingual corpora



No bilingual resource

Chinese monolingual corpora

Arabic embeddings

Chinese embeddings

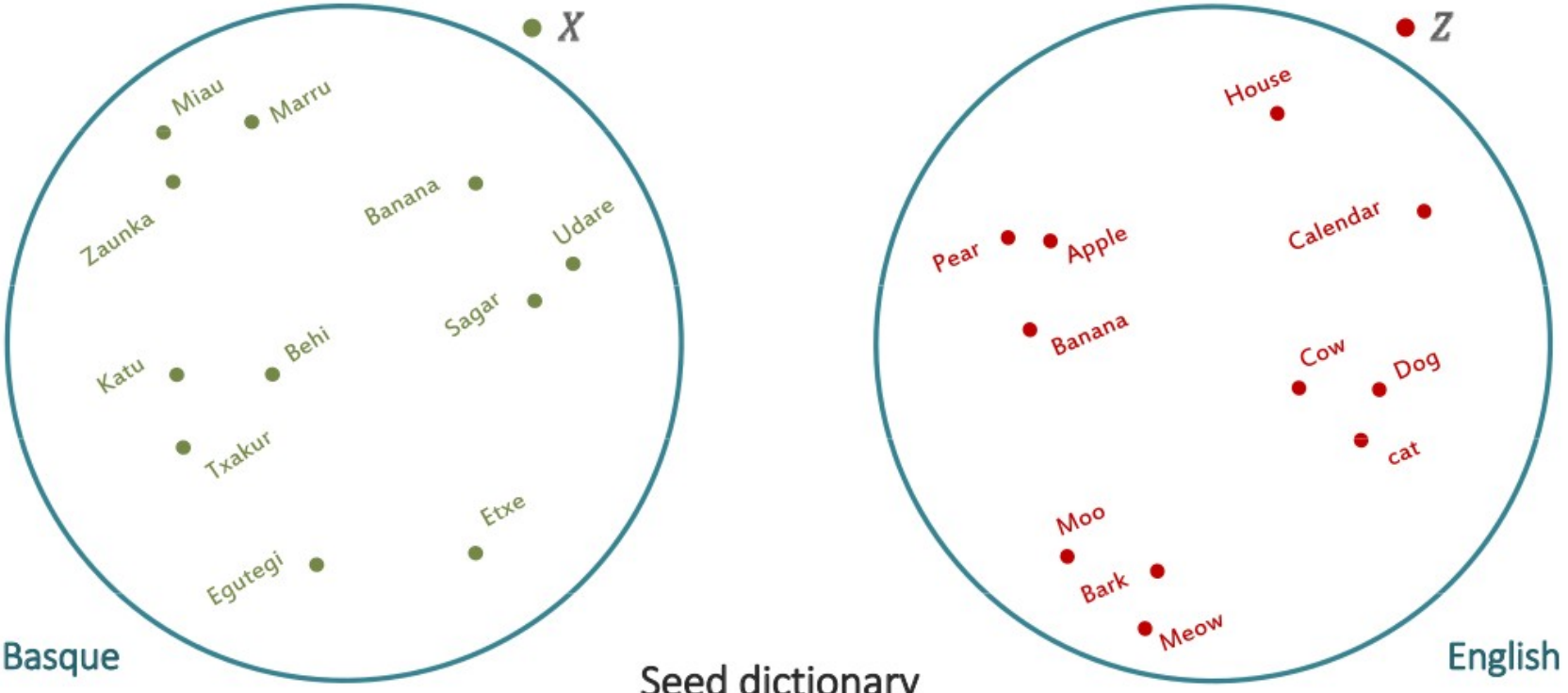
Bilingual embeddings

Bilingual dictionaries

Crosslingual & multilingual applications

Machine translation

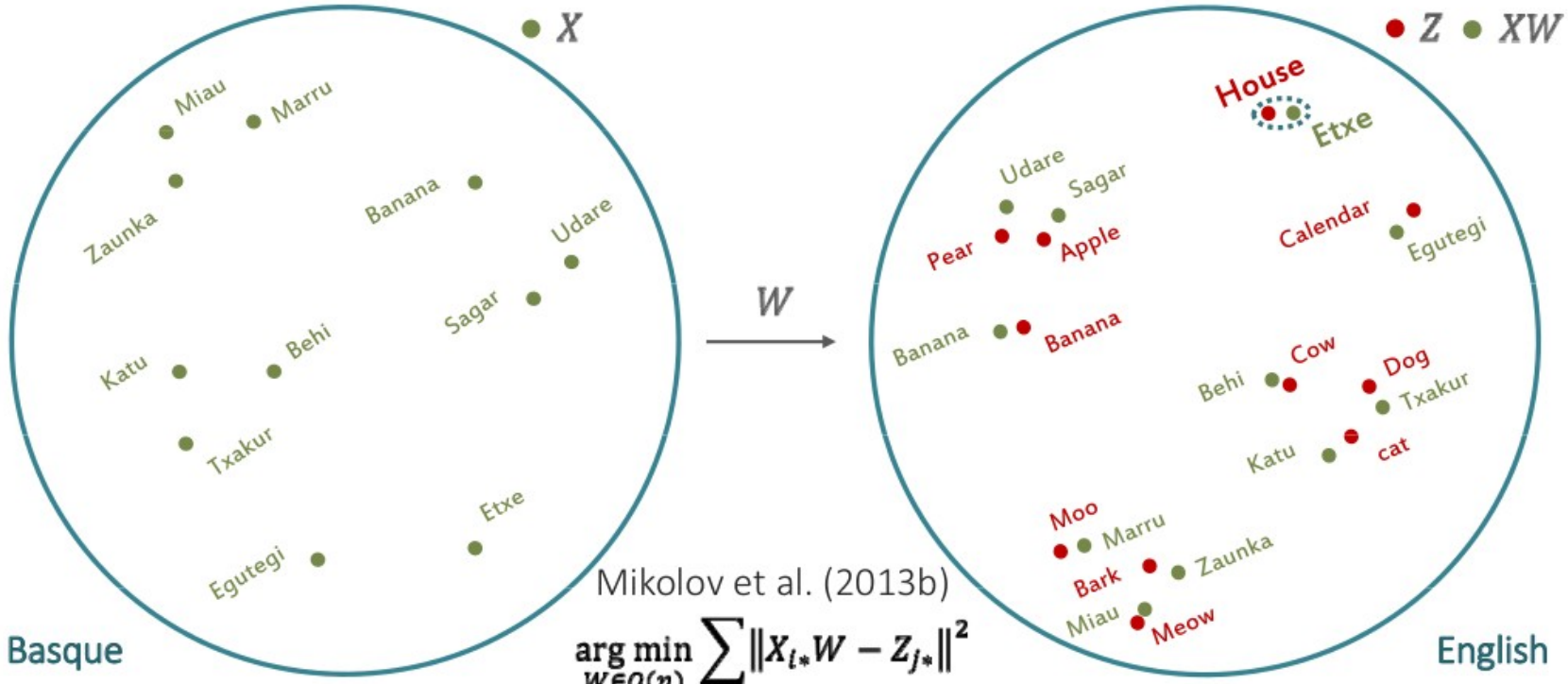
Cross-lingual embeddings



Seed dictionary

Txakur	Dog
Sagar	Apple
⋮	⋮
Egutegi	Calendar

Cross-lingual word embeddings

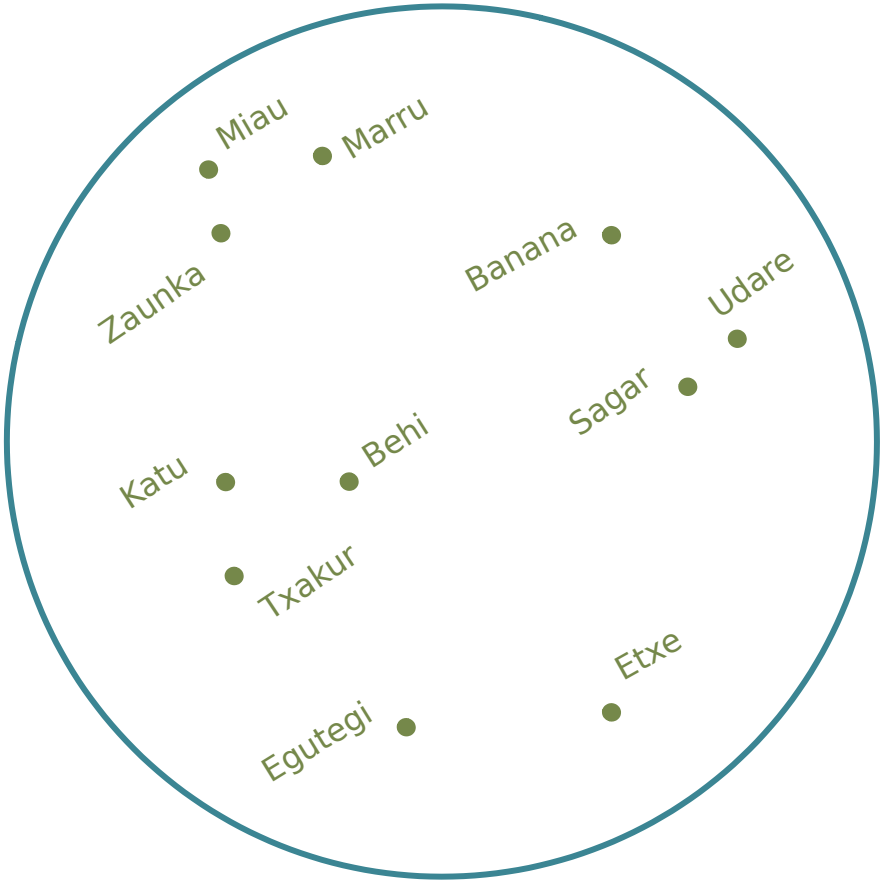


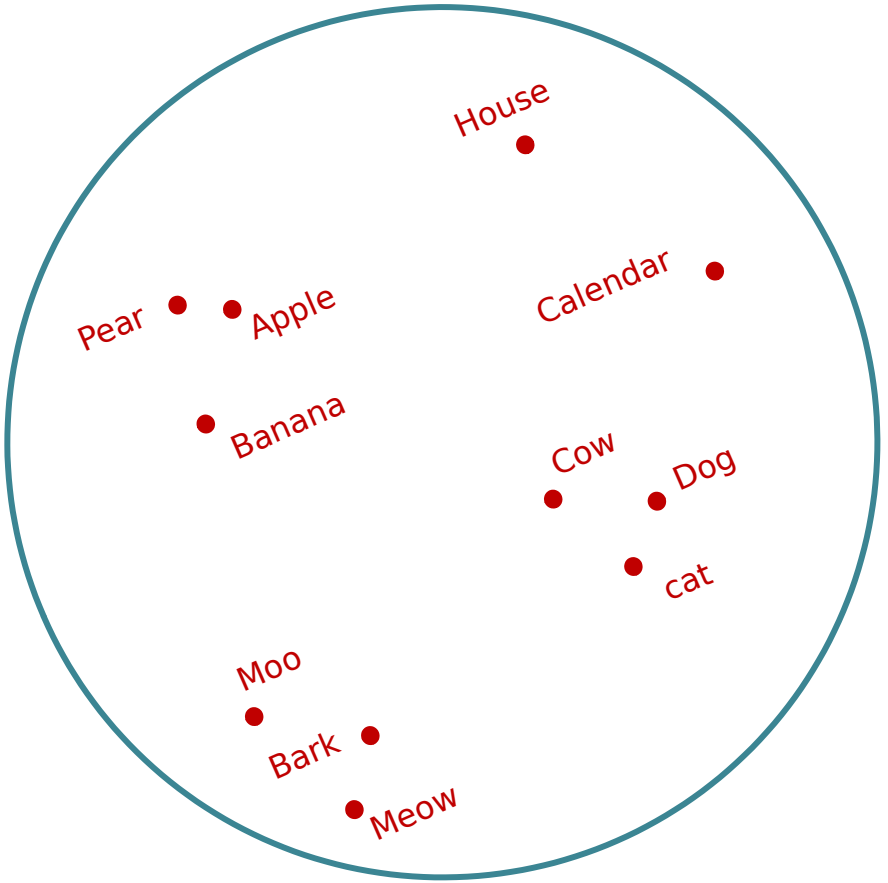
Mikolov et al. (2013b)

$$\arg\min_{W \in O(n)} \sum_{i} \|X_{i*}W - Z_{j*}\|^2$$

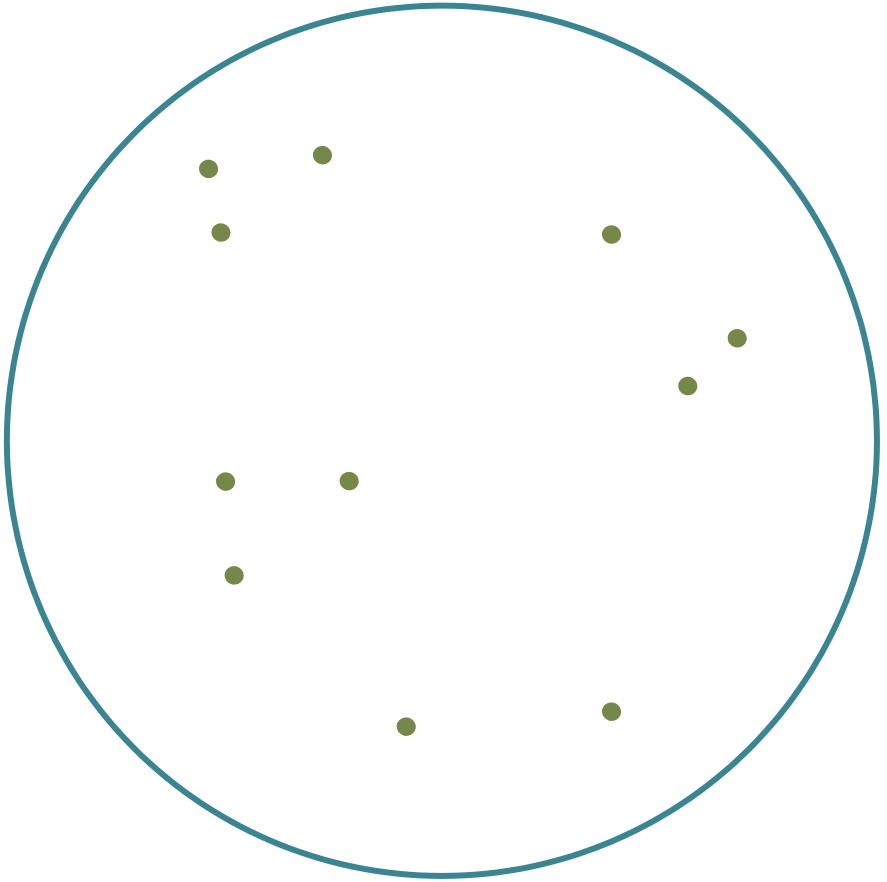
$$\begin{matrix} \text{Txakur} \\ \text{Sagar} \\ \vdots \\ \text{Egutegi} \end{matrix} \begin{bmatrix} X_{1,*} \\ X_{2,*} \\ \vdots \\ X_{n,*} \end{bmatrix} [W] \approx \begin{bmatrix} Z_{1,*} \\ Z_{2,*} \\ \vdots \\ Z_{n,*} \end{bmatrix} \begin{matrix} \text{Dog} \\ \text{Apple} \\ \vdots \\ \text{Calendar} \end{matrix}$$

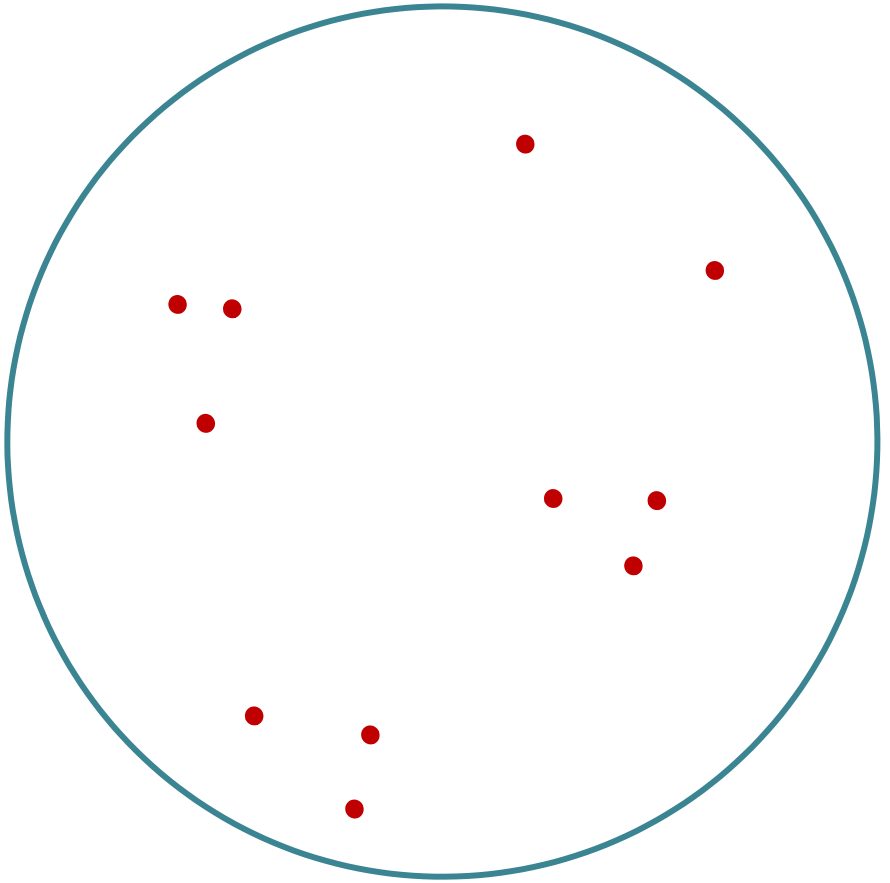
How can it work without dictionaries?



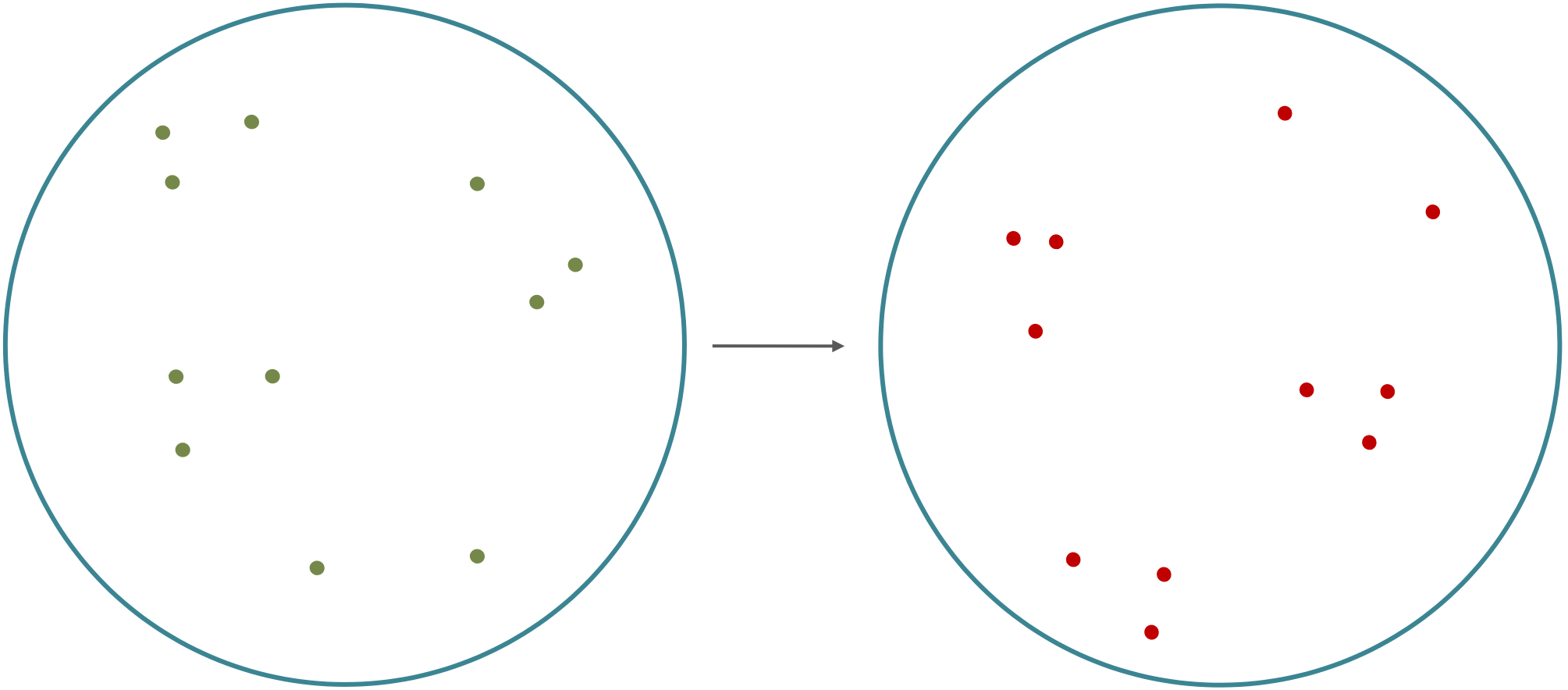


How can it work without dictionaries?

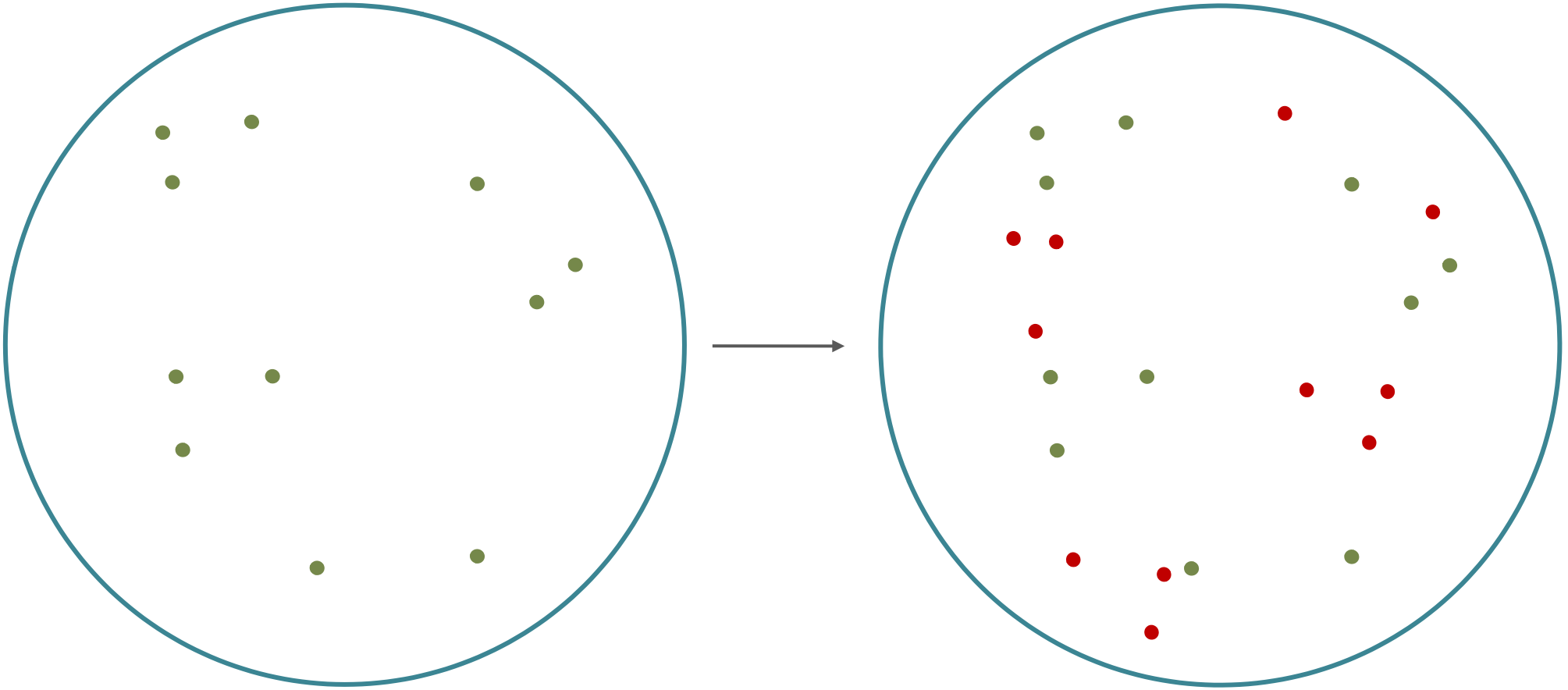




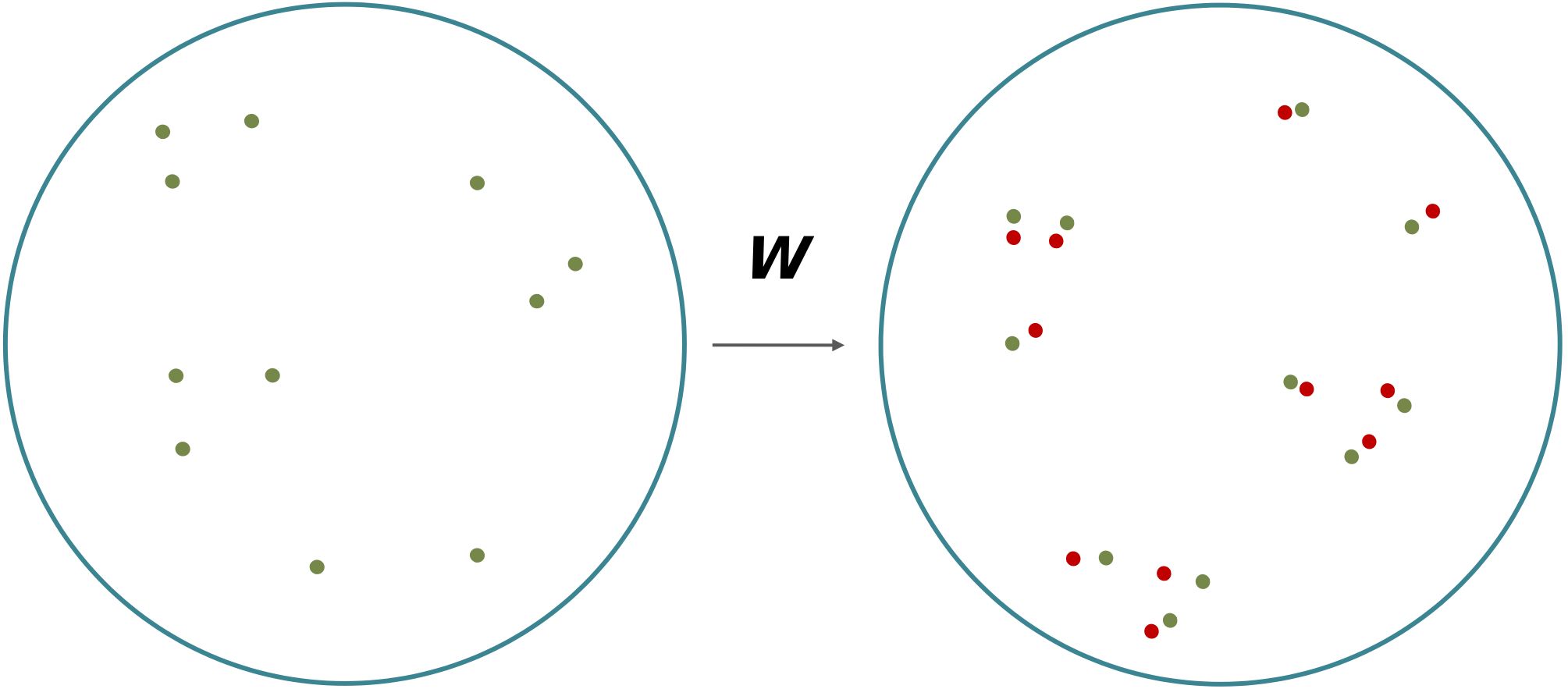
How can it work without dictionaries?



How can it work without dictionaries?

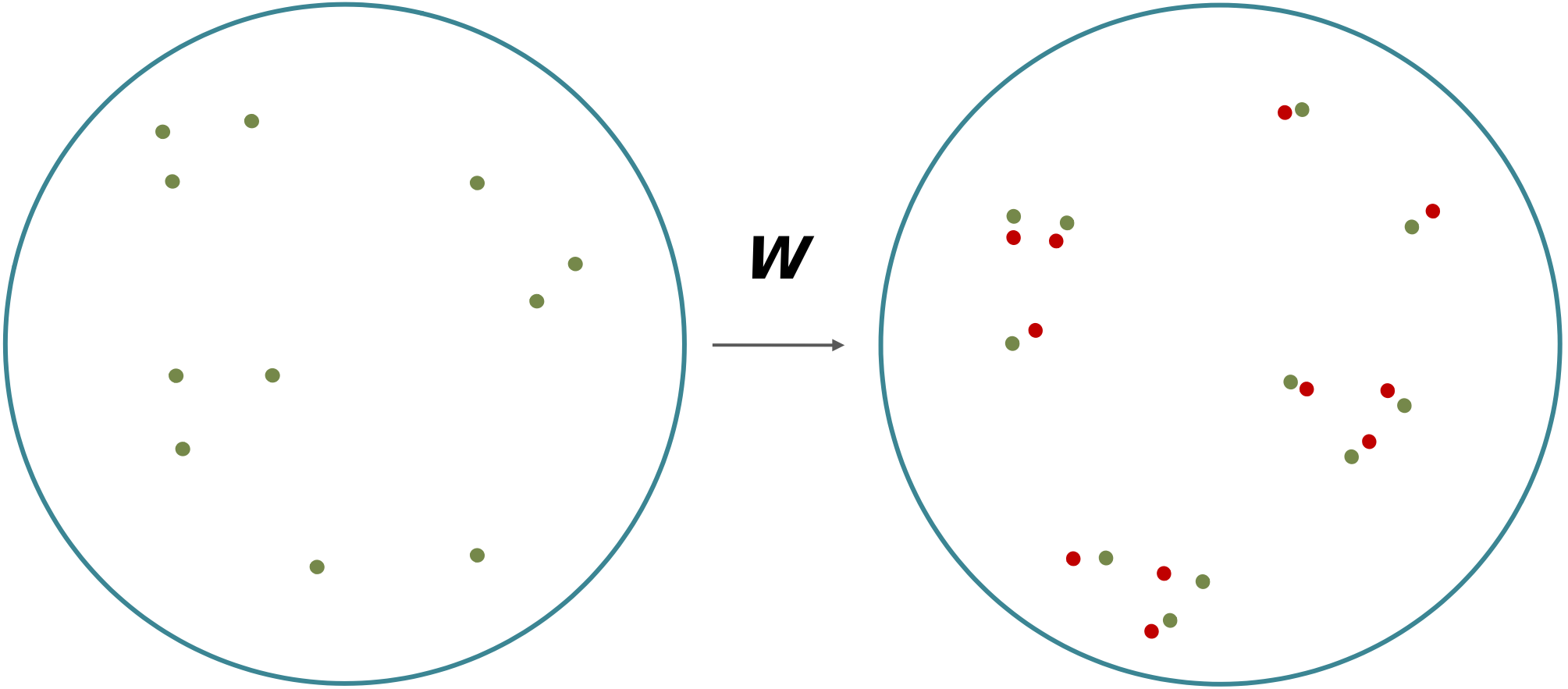


How can it work without dictionaries?



How can it work without dictionaries?

Languages are (to a large extent)
isometric in word embedding space (!)



Contents

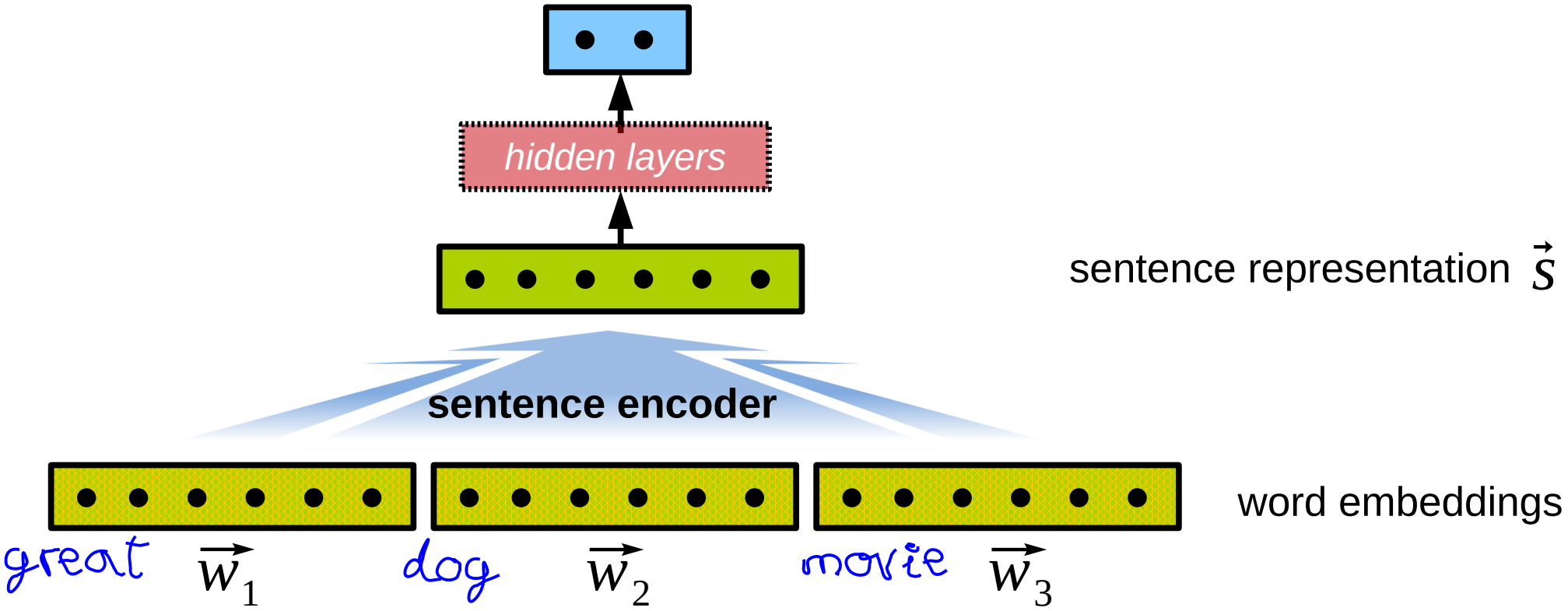
- Introduction to NLP
  - Deep Learning ~ Learning Representations
- Text as bag of words
  - Text Classification
  - Representation learning and word embeddings
  - Superhuman: xlingual word embeddings https://www.aclweb.org/anthology/P18-1073/
- Text as a sequence RNN (LSTM), attention, transformers
  - Seq2seq, Machine Translation
  - Representation learning and pre-trained language models
  - Superhuman: unsupervised MT https://www.aclweb.org/anthology/P19-1019/

Sentence encoder

A function:
input a sequence of word embeddings
output a sentence representation

$\vec{w}_i \in \mathbb{R}^D$

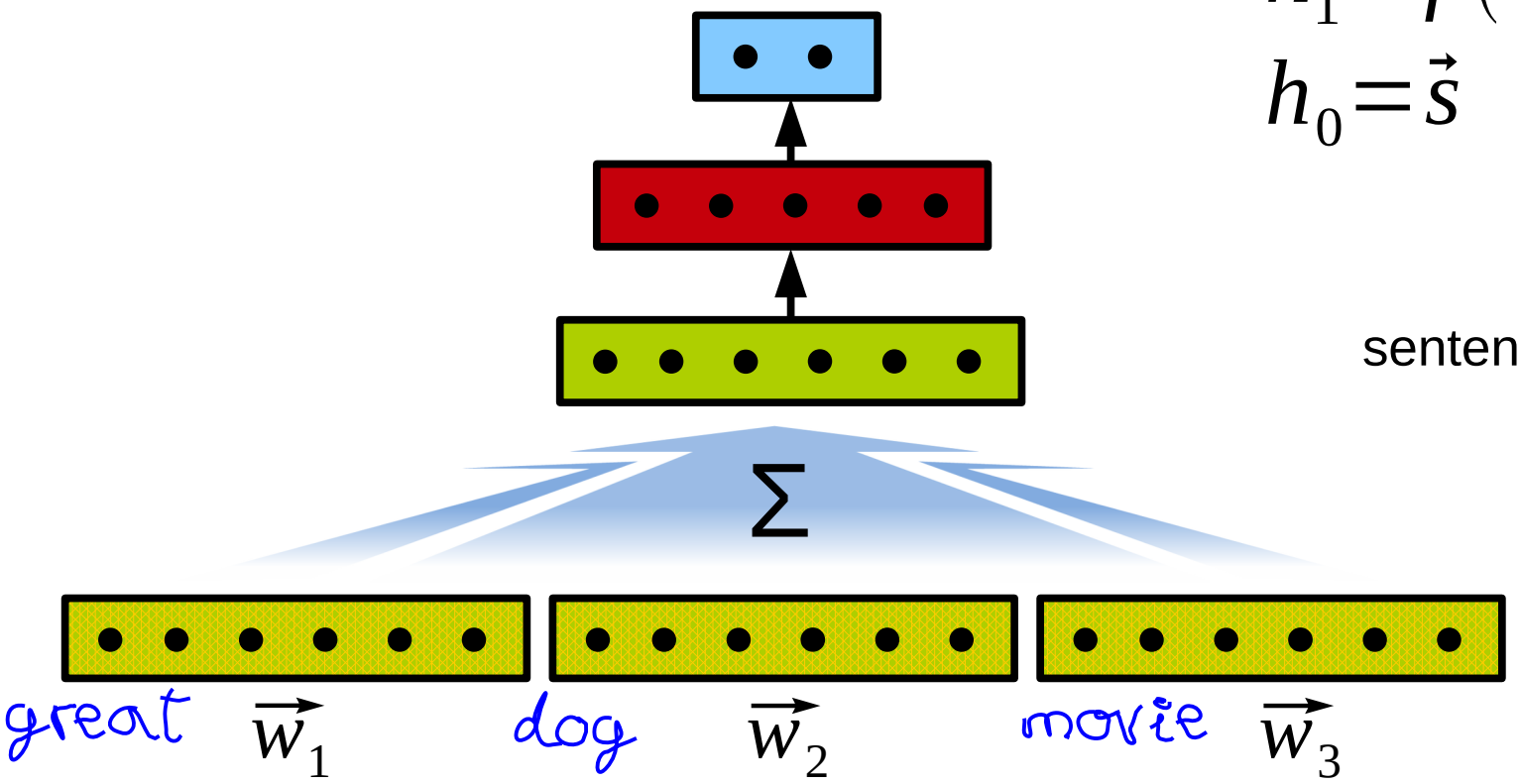
$\vec{s} \in \mathbb{R}^{D'}$



Sentence encoder

Baseline 1: Continuous bag of words

$$h_1 = f(W_1 h_0 + b_1)$$
$$h_0 = \vec{s}$$



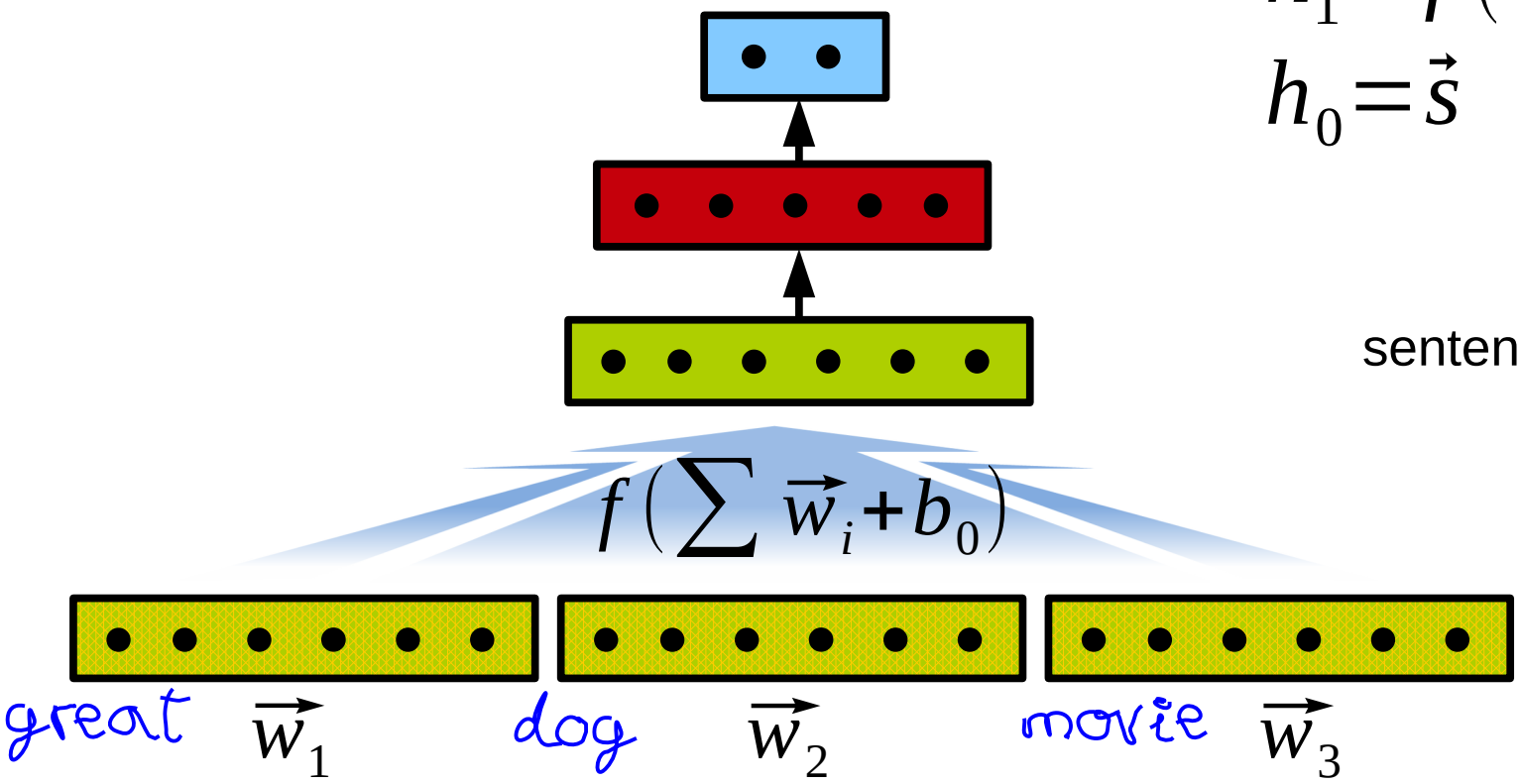
sentence representation $\vec{s}$

word embeddings

Sentence encoder

Baseline 1': MLP
It is implicitly encoding sequences as bag of words

$$h_1 = f(W_1 h_0 + b_1)$$
$$h_0 = \vec{s}$$



$f(\sum \vec{w}_i + b_0)$

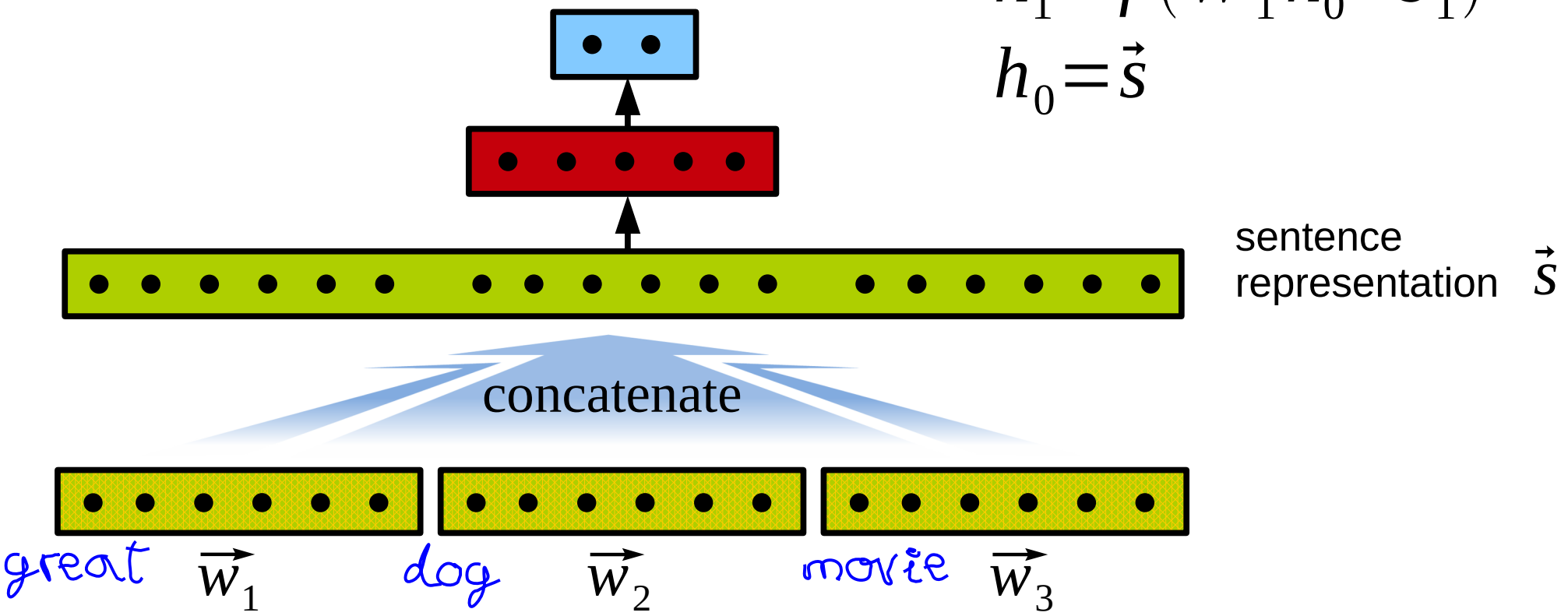
$\vec{w}_1$ $\vec{w}_2$ $\vec{w}_3$

sentence representation $\vec{s}$

Sentence encoder

Baseline 2: Add word order with concatenation

$$h_1 = f(W_1 h_0 + b_1)$$
$$h_0 = \vec{s}$$



Sentence encoder

Baseline 2: Add word order with concatenation

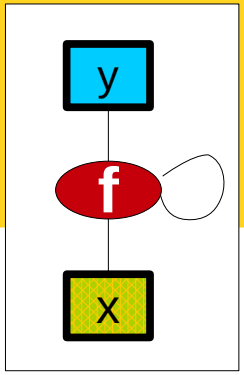
$$h_1 = f\left(W_1 h_0 + b_1\right)$$
$$h_0 = \vec{s}$$

It is a masterful performance but, for me, not enough to make me a fan of the film.

It is not only lovely to look at (the exquisite northern Italian countryside made me want to hop on a plane that moment), but is made even better by the subtle performance of Timothée Chalamet. "

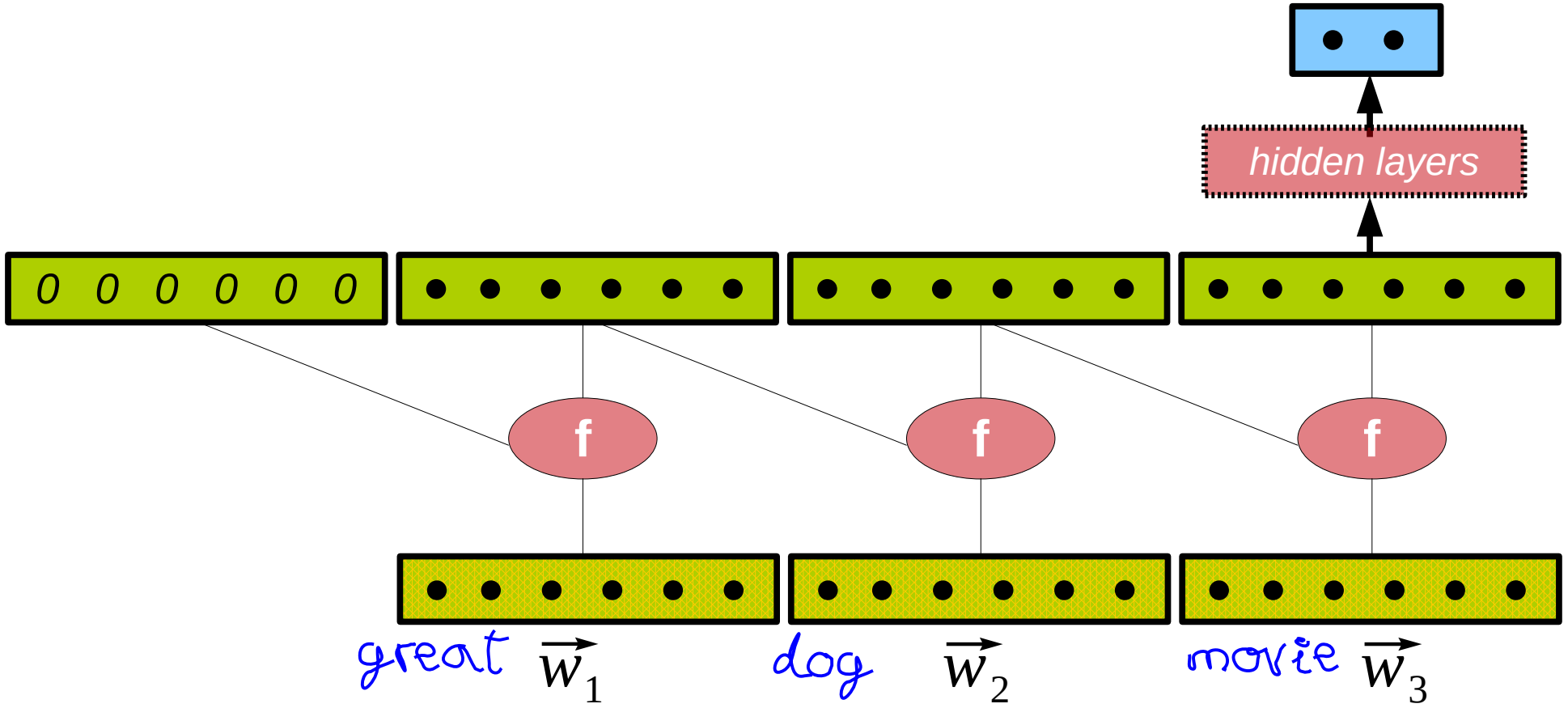
Fantastic movie

Sentence encoder

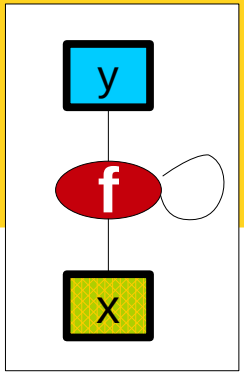


Recurrent Neural Network (RNN)

Apply a single function f recursively



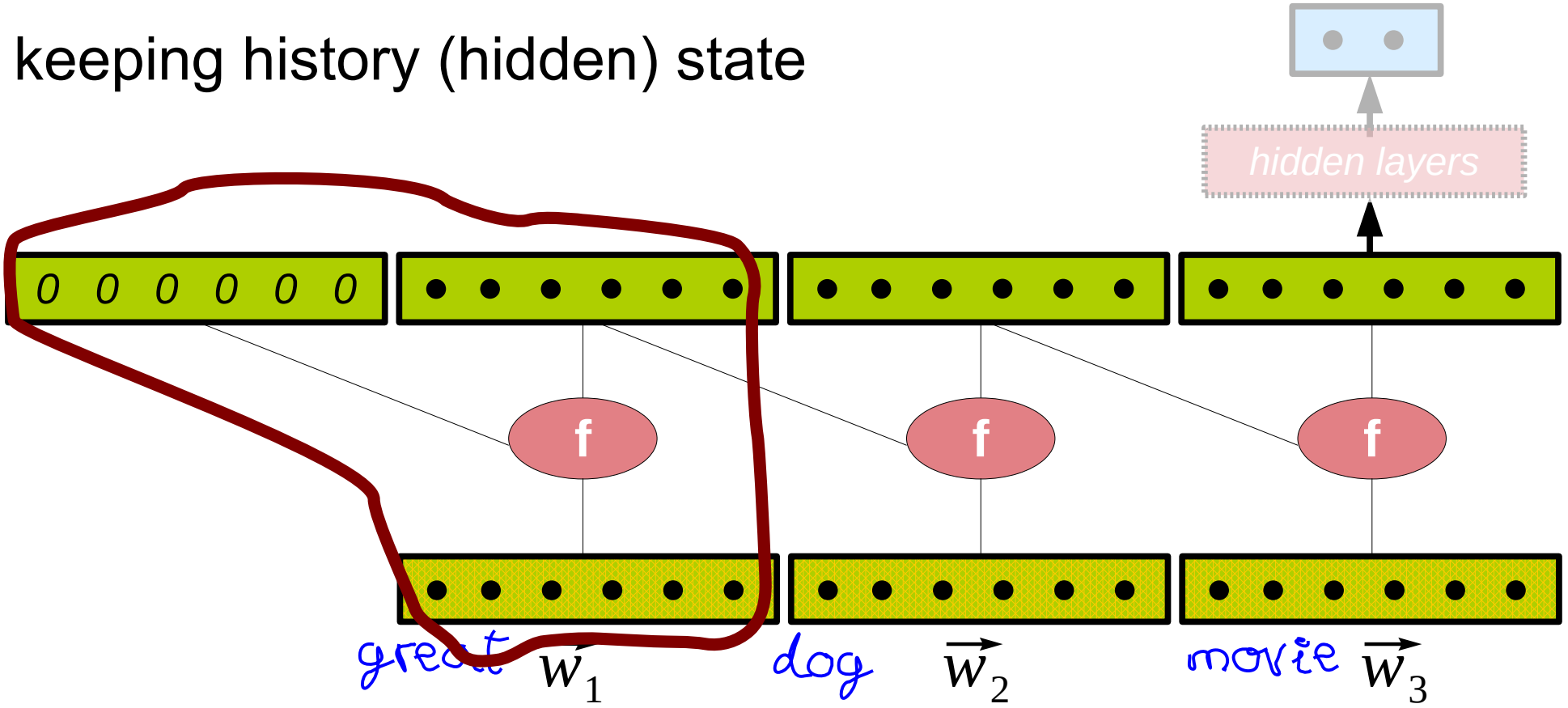
Sentence encoder: RNN



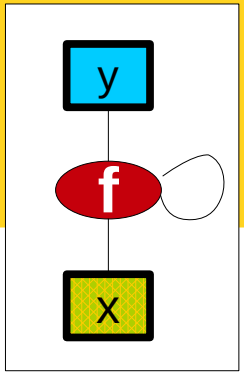
Recurrent Neural Network (RNN)

Apply a single function f recursively

keeping history (hidden) state



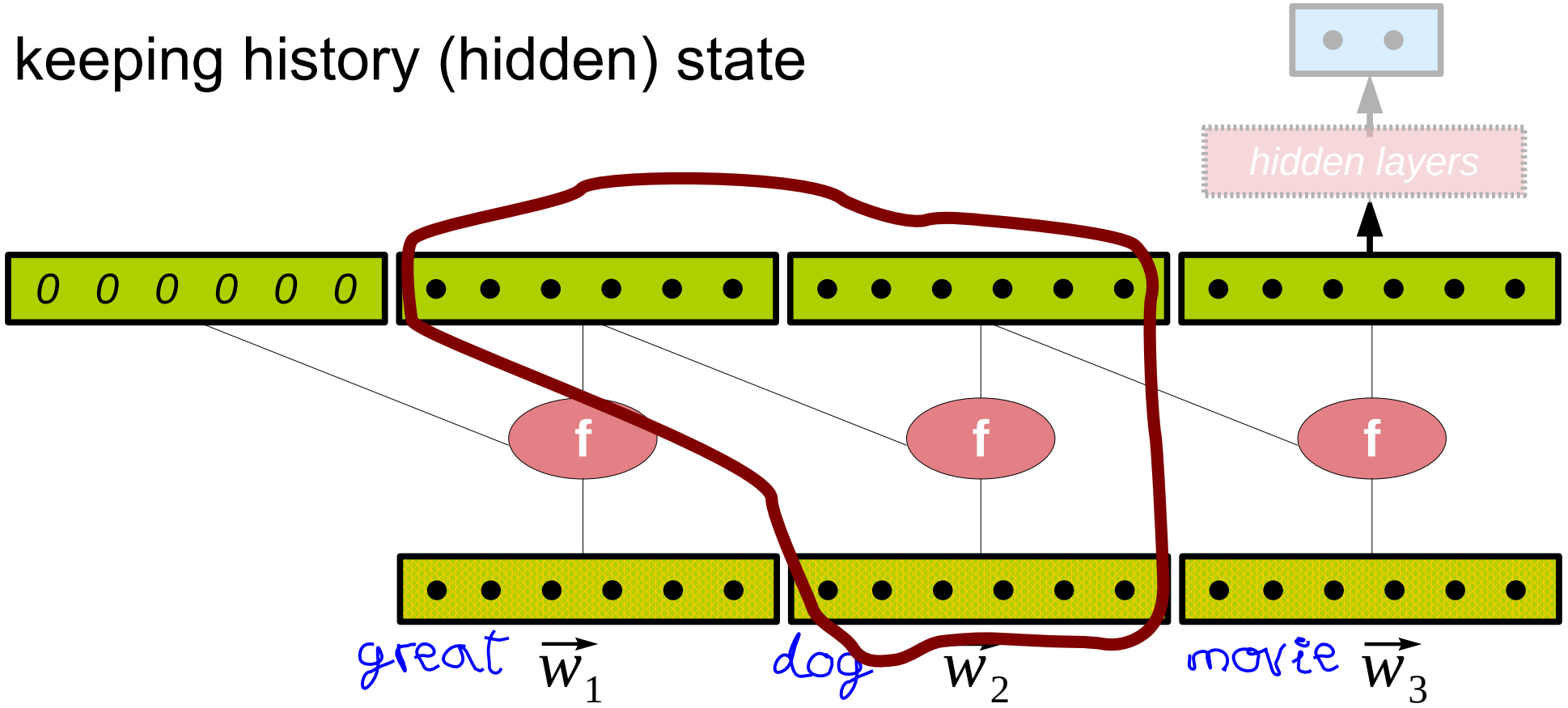
Sentence encoder: RNN



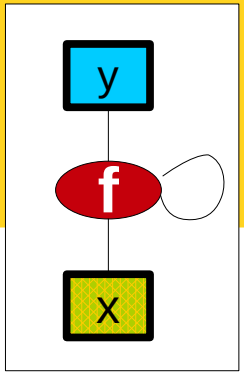
Recurrent Neural Network (RNN)

Apply a single function f recursively

keeping history (hidden) state



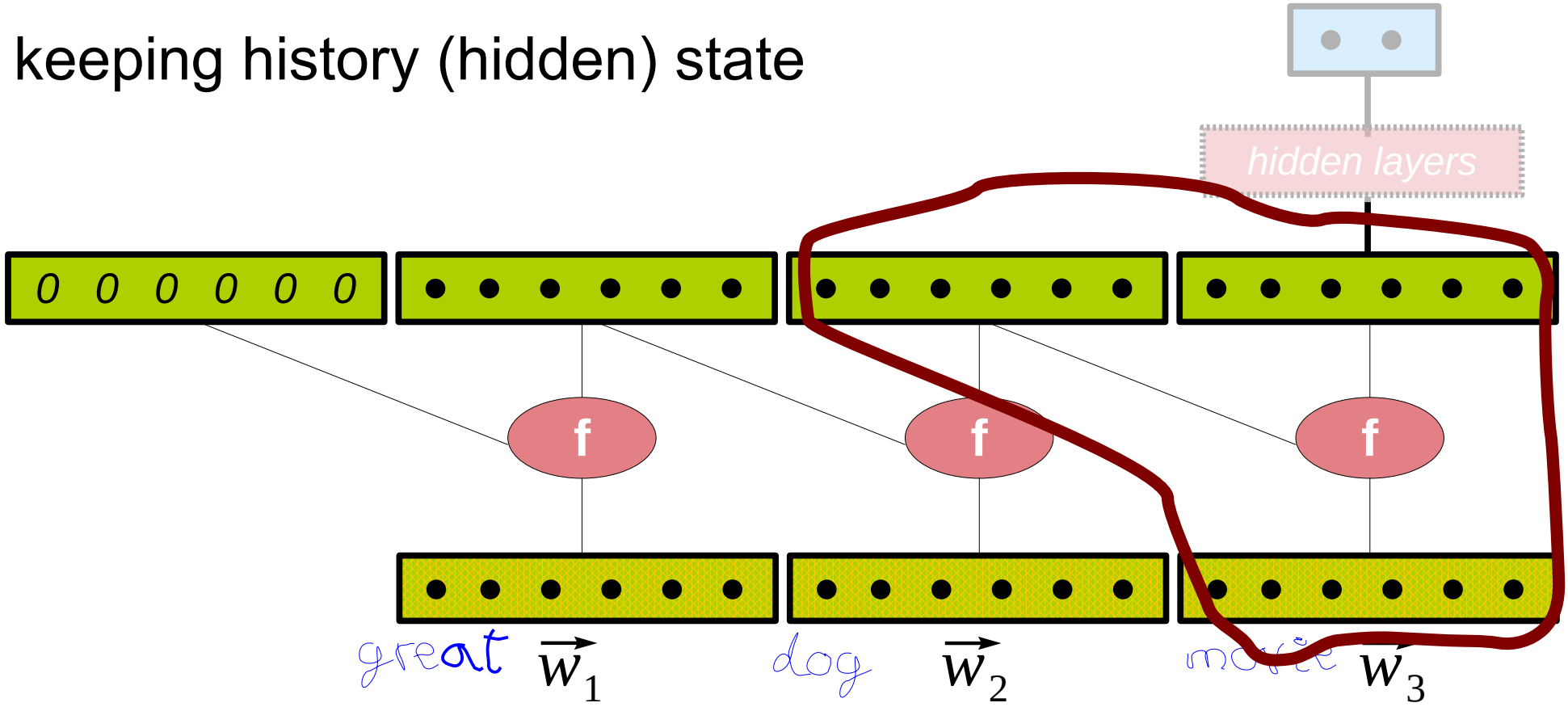
Sentence encoder: RNN



Recurrent Neural Network (RNN)

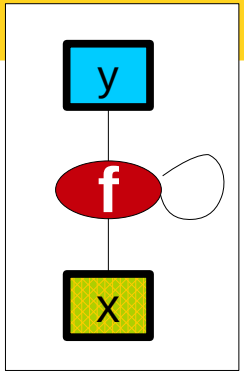
Apply a single function f recursively

keeping history (hidden) state

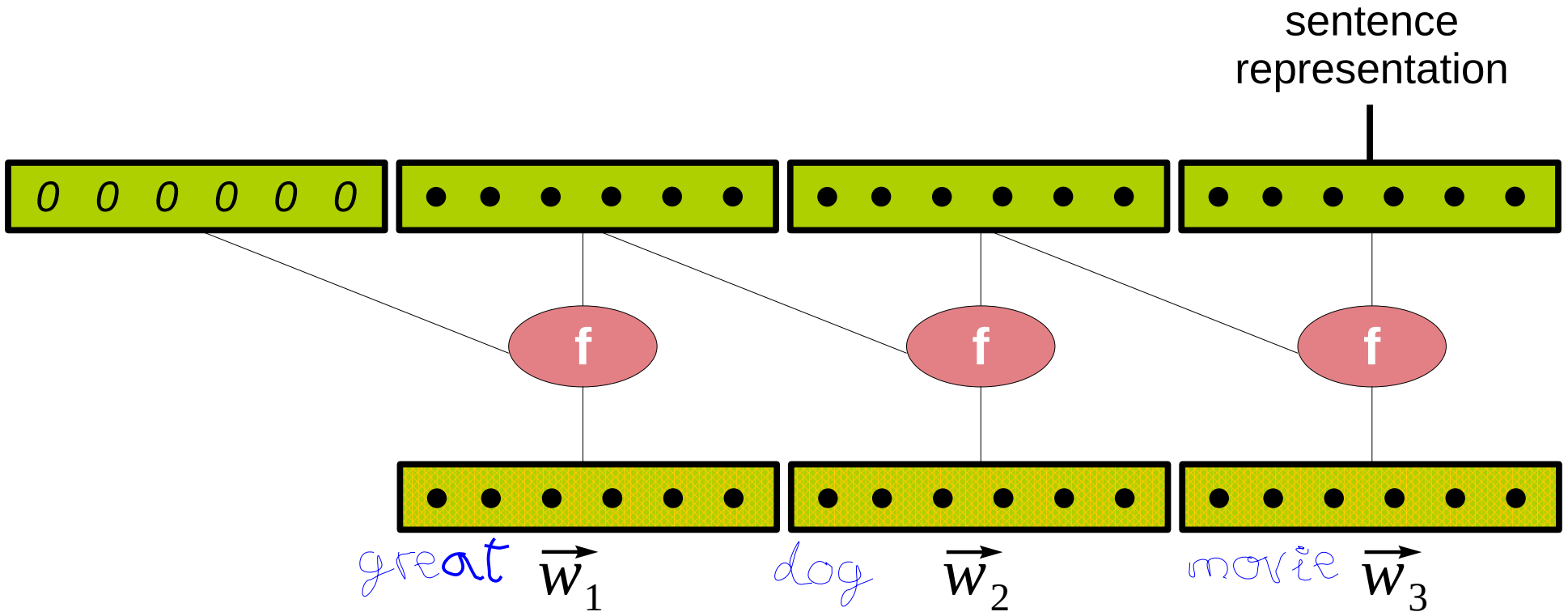


Sentence encoder: Basic RNN

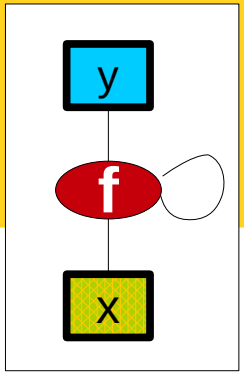
Apply a single function **f** recursively.



$$\vec{h}_t = f(\vec{h}_{t-1}, \vec{w}_t) = \tanh(W[\vec{h}_{t-1}, \vec{w}_t] + \vec{b})$$

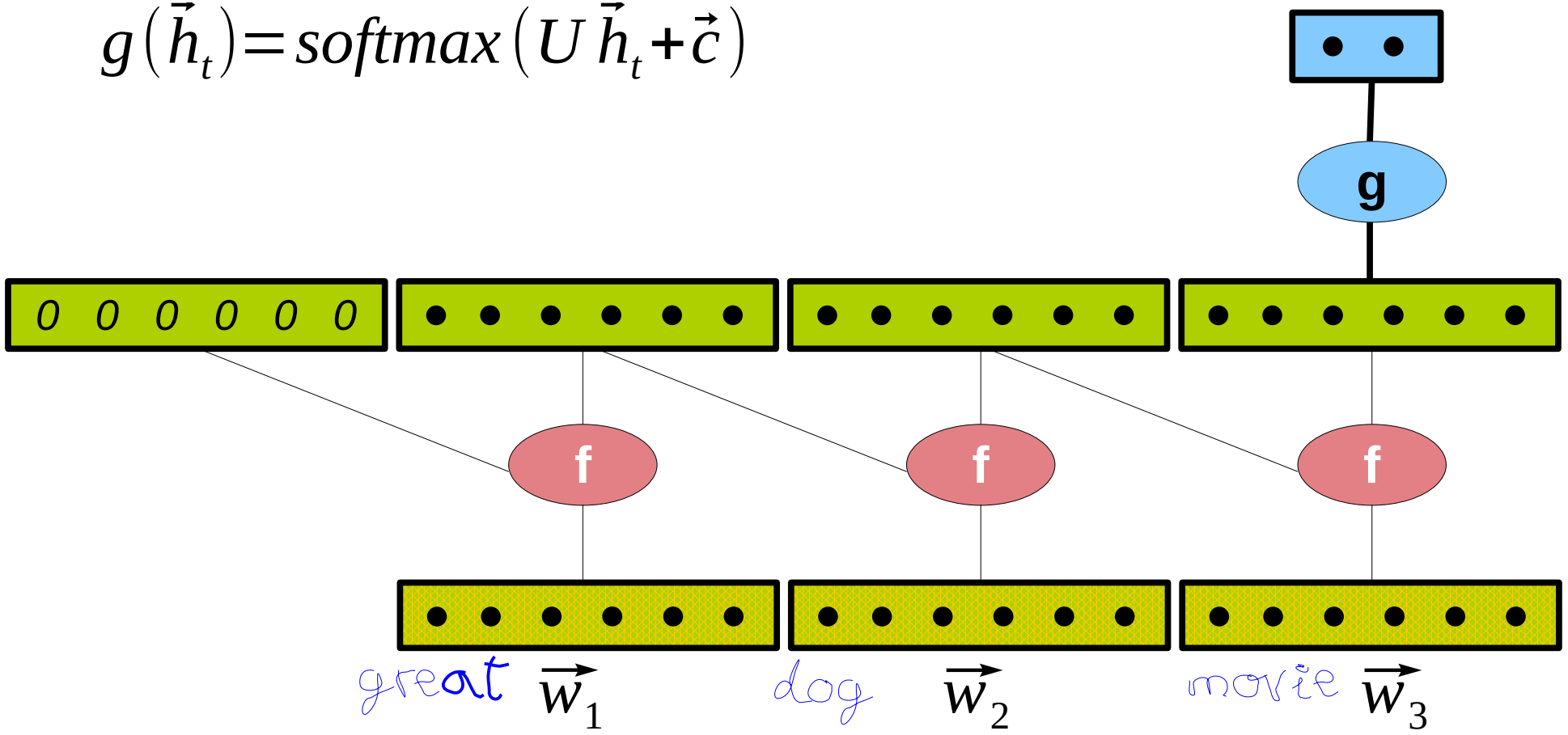


RNN: classifier

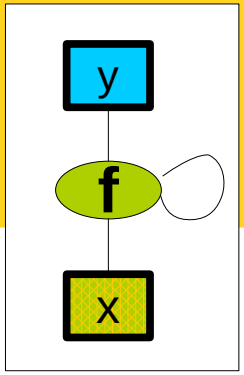


- Last hidden state is input to classifier

$$g(\vec{h}_t) = softmax(U\vec{h}_t + \vec{c})$$

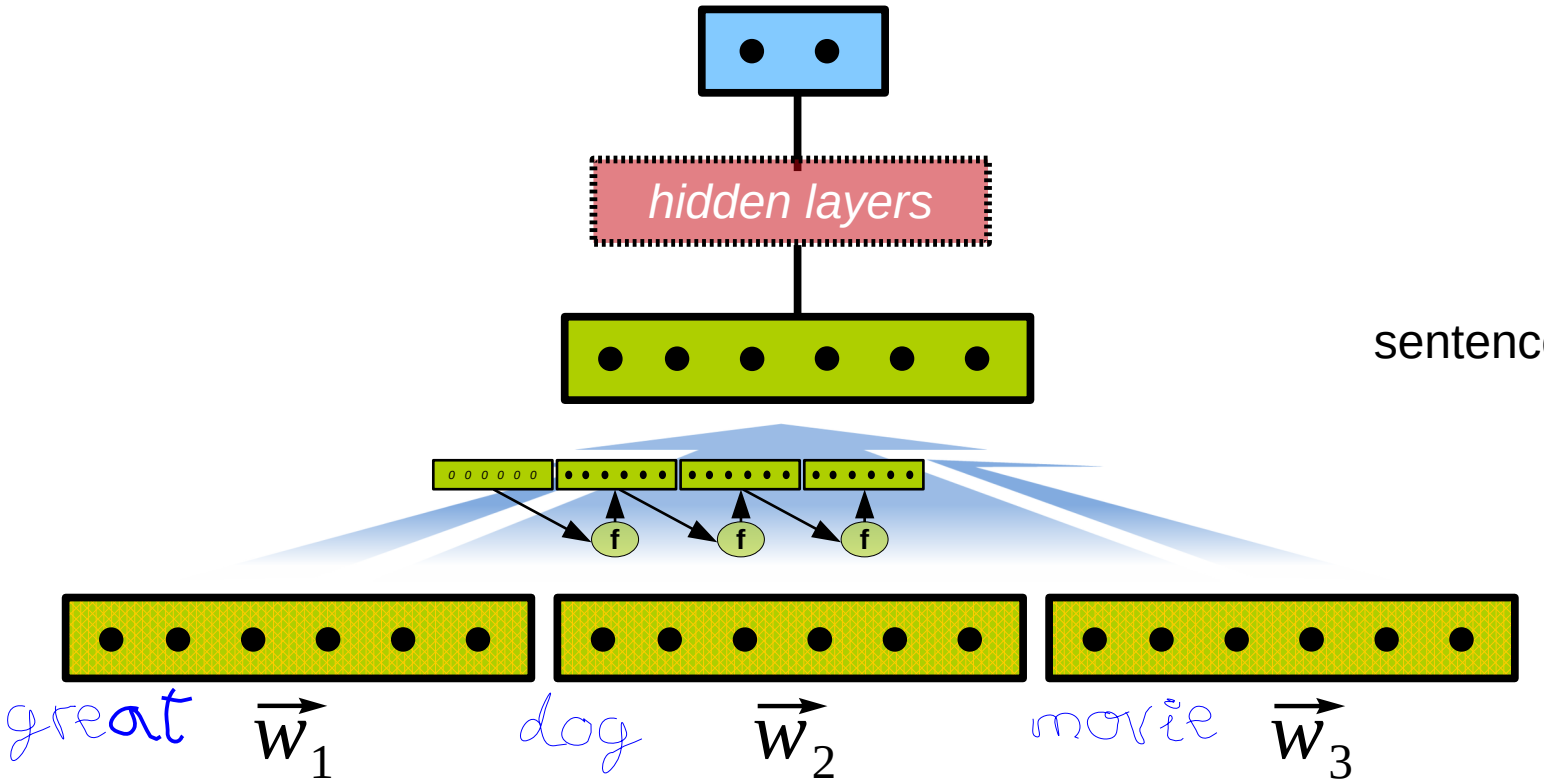


RNN as sentence encoder



Sentence encoder 3: RNN

$$\vec{h}_t = \tanh\left(W[\vec{h}_{t-1}, \vec{w}_t] + \vec{b}\right)$$



sentence representation $\vec{s}$

word embeddings

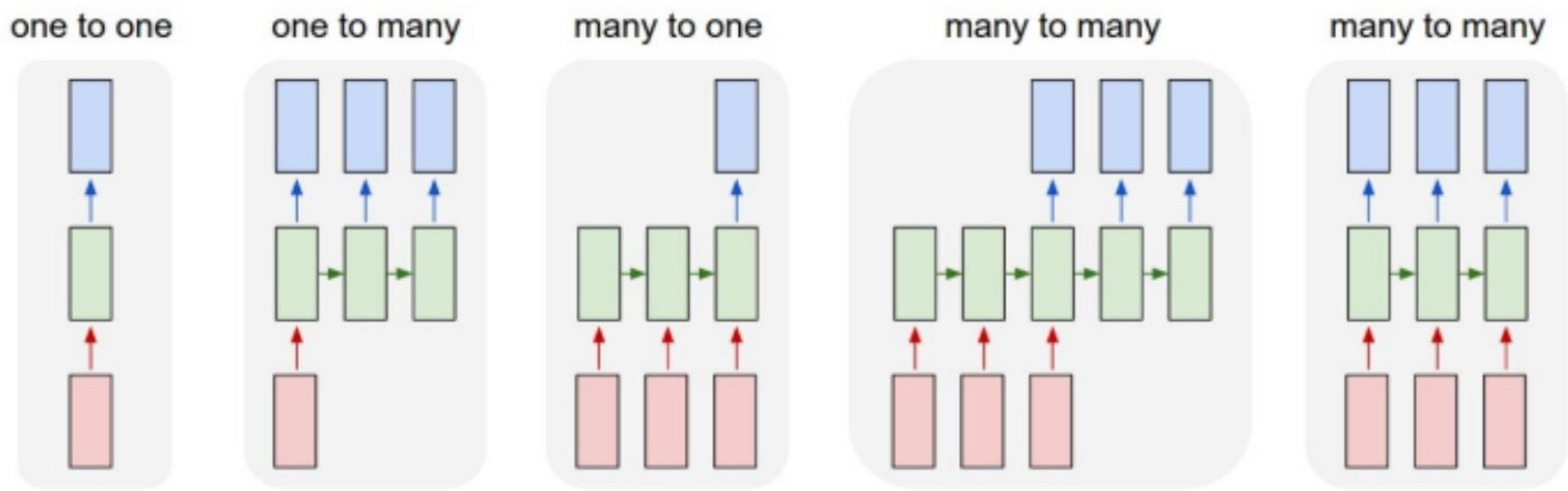
Sequente to sequence models

- For any problem that can be interpreted as a transformation from one sequence to another:
  - Model it as a **pair of RNNs:**
  - An **encoder** that reads the input sentence, outputs nothing
  - A **decoder** whose starting hidden state is the last hidden state of the encoder, and that generates a sentence (RNN language model)
  - Give it lots of data….
    Success is guaranteed (Sutskever et al. 2014)

Sequente to sequence models

- They are state-of-the-art in many problems, some of them fairly novel
  - Speech → text (and viceversa)
  - Foreign sentences → translation
  - Emails → simple replies (Gmail)
  - Python functions → result
  - English sentences → parsing instructions
  - Question → answer (Chatbots, Google Duplex)
  - Image → textual description
  - Video → textual descriptions
  - ...

RNNs offer a lot of flexibility



MLP

Source: Fei-Fei Li & Andrej Karpathy and Justin Johnson

RNNs offer a lot of flexibility

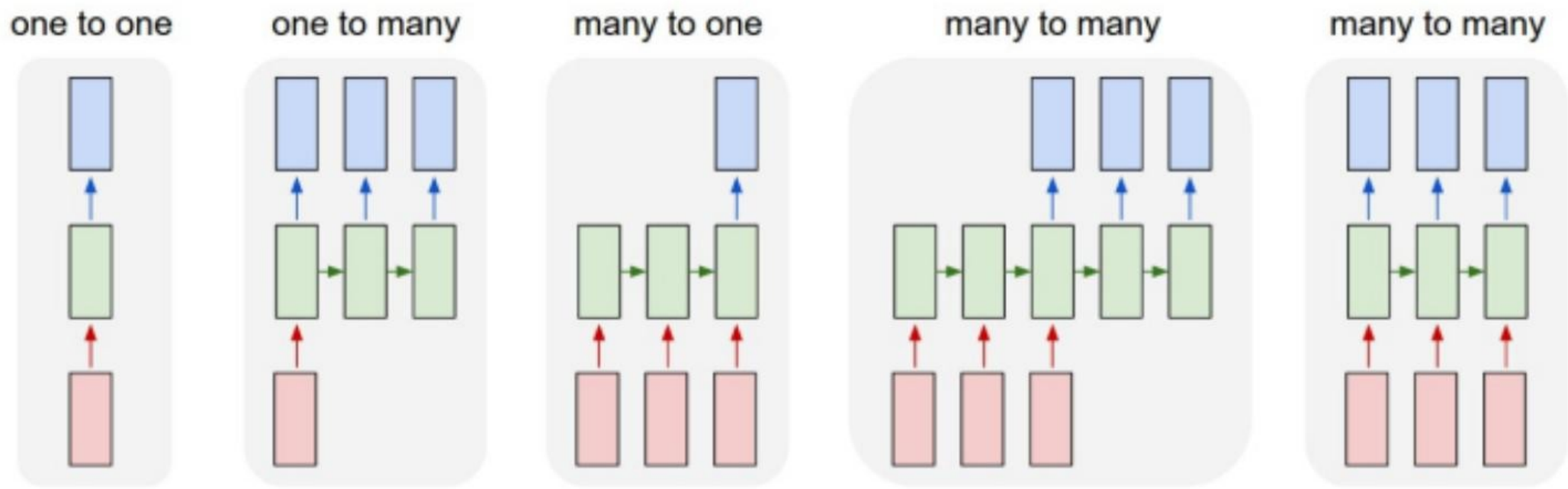
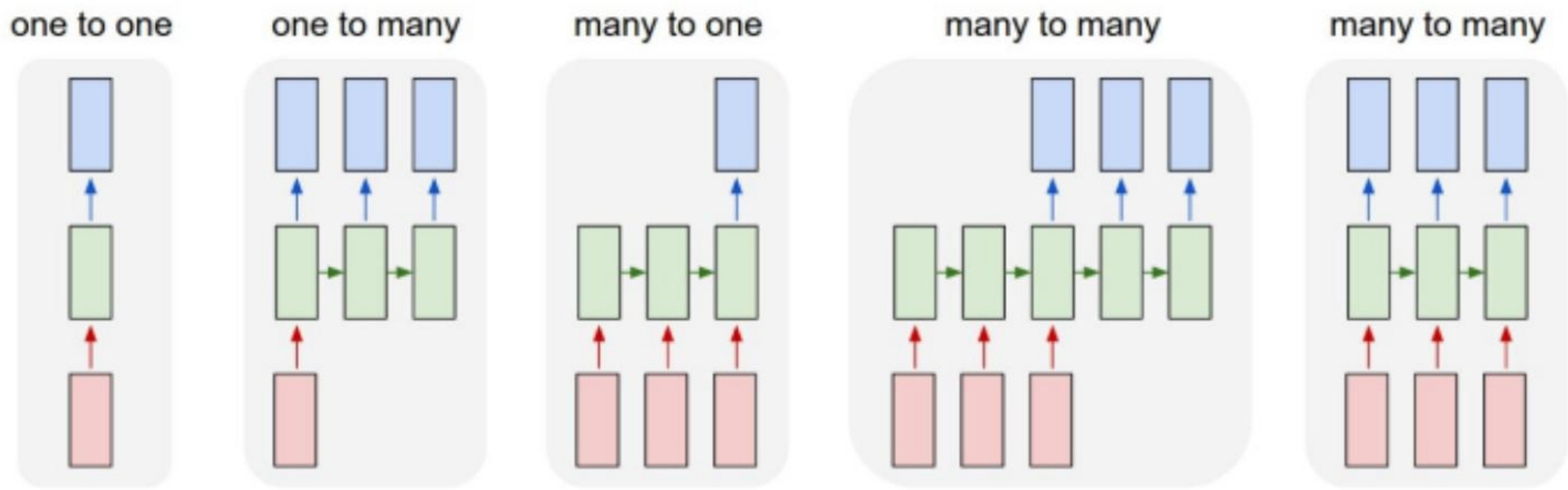


Image Captioning

Source: Fei-Fei Li & Andrej Karpathy and Justin Johnson

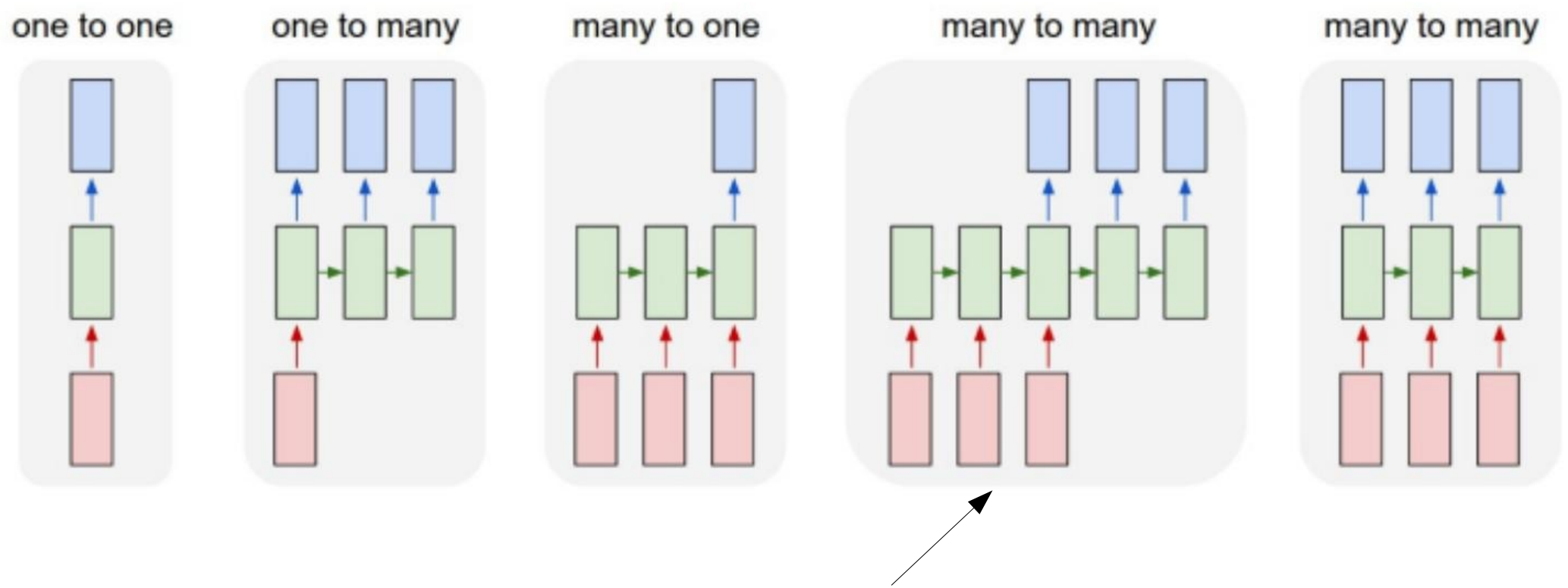
RNNs offer a lot of flexibility



Sentiment classification

Source: Fei-Fei Li & Andrej Karpathy and Justin Johnson

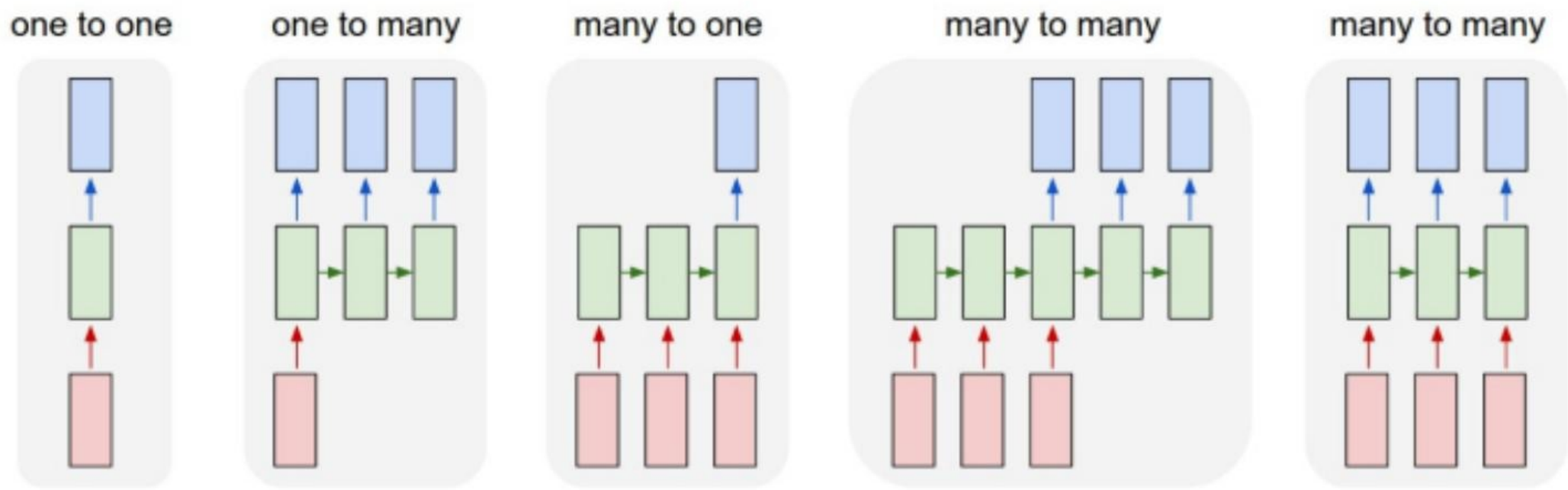
RNNs offer a lot of flexibility



Machine translation

Source: Fei-Fei Li & Andrej Karpathy and Justin Johnson

RNNs offer a lot of flexibility



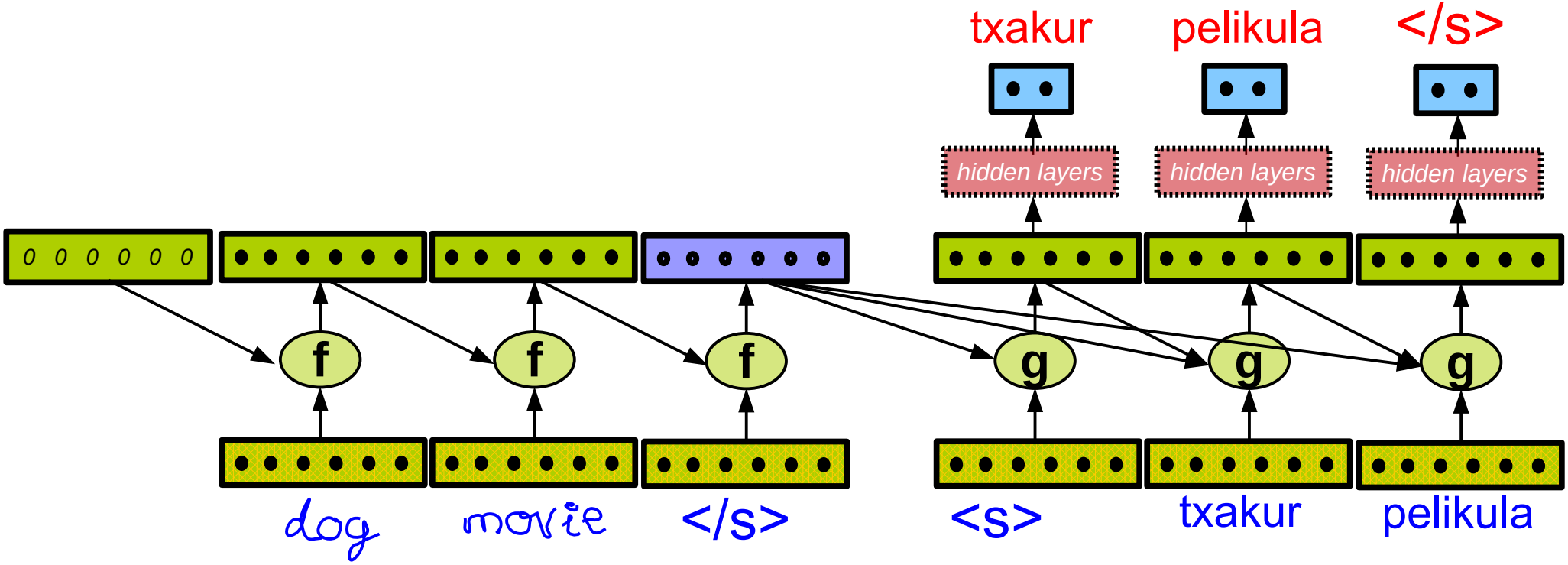
Video classification at frame level

Source: Fei-Fei Li & Andrej Karpathy and Justin Johnson

Sequence to sequence for MT

Combine two RLM: encoder and decoder

Train as regular RLM



Sequence to sequence for MT

Combine two RLM

Train as a regular RLM

- Note that decoder is conditioned on last hidden state from encoder:

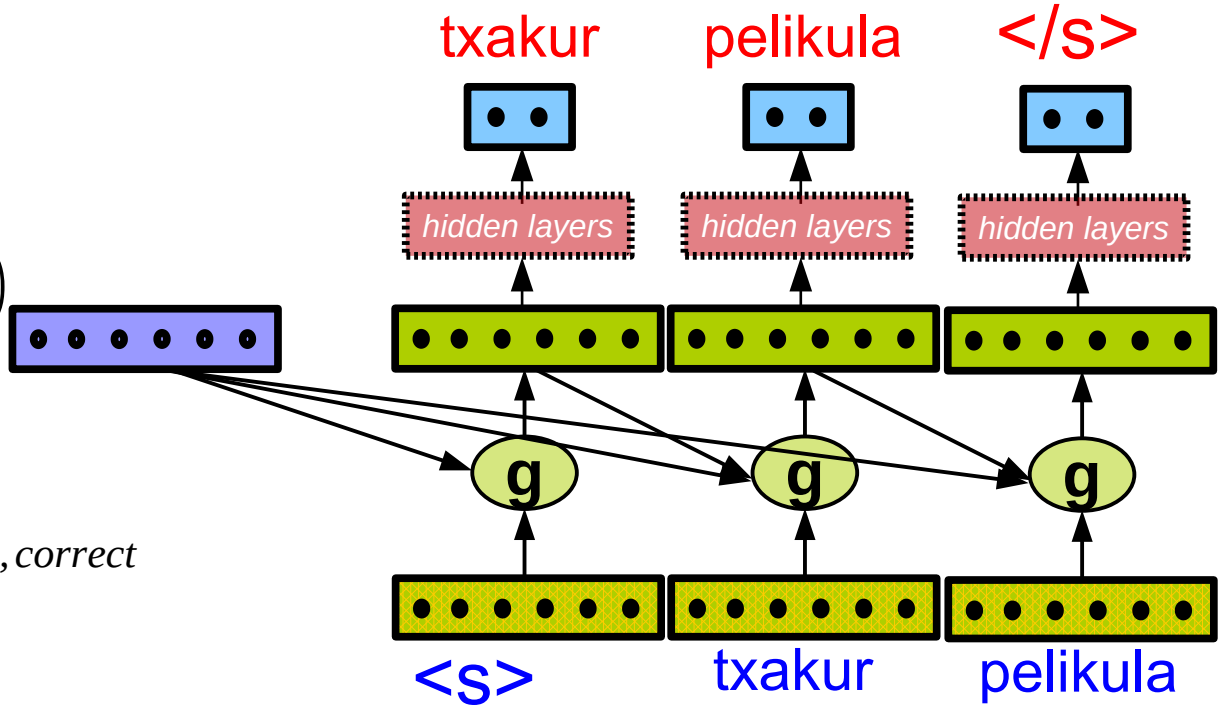
$$C = \vec{h}_{encoder}$$

$$\vec{h}_t = \tanh\left(W\left[\vec{h}_{t-1}, \vec{w}_t, C\right] + \vec{b}\right)$$

$$\hat{y}_t = softmax\left(W^S \vec{h}_t + \vec{c}\right)$$

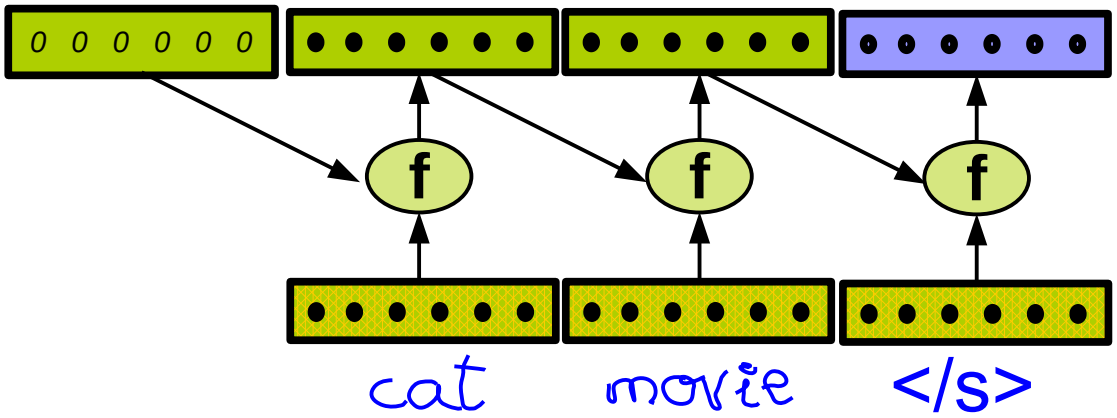
$$p\left(w_1, \ldots, w_m | \boldsymbol{h}_{\mathbf{encoder}}\right) \approx \prod_{t=0}^{m-1} \hat{y}_{t,correct}$$

$$J_{\text{sentence}} = -\frac{1}{T}\sum_{t=1}^{m} \log \hat{y}_{t,correct}$$



Sequence to sequence for MT

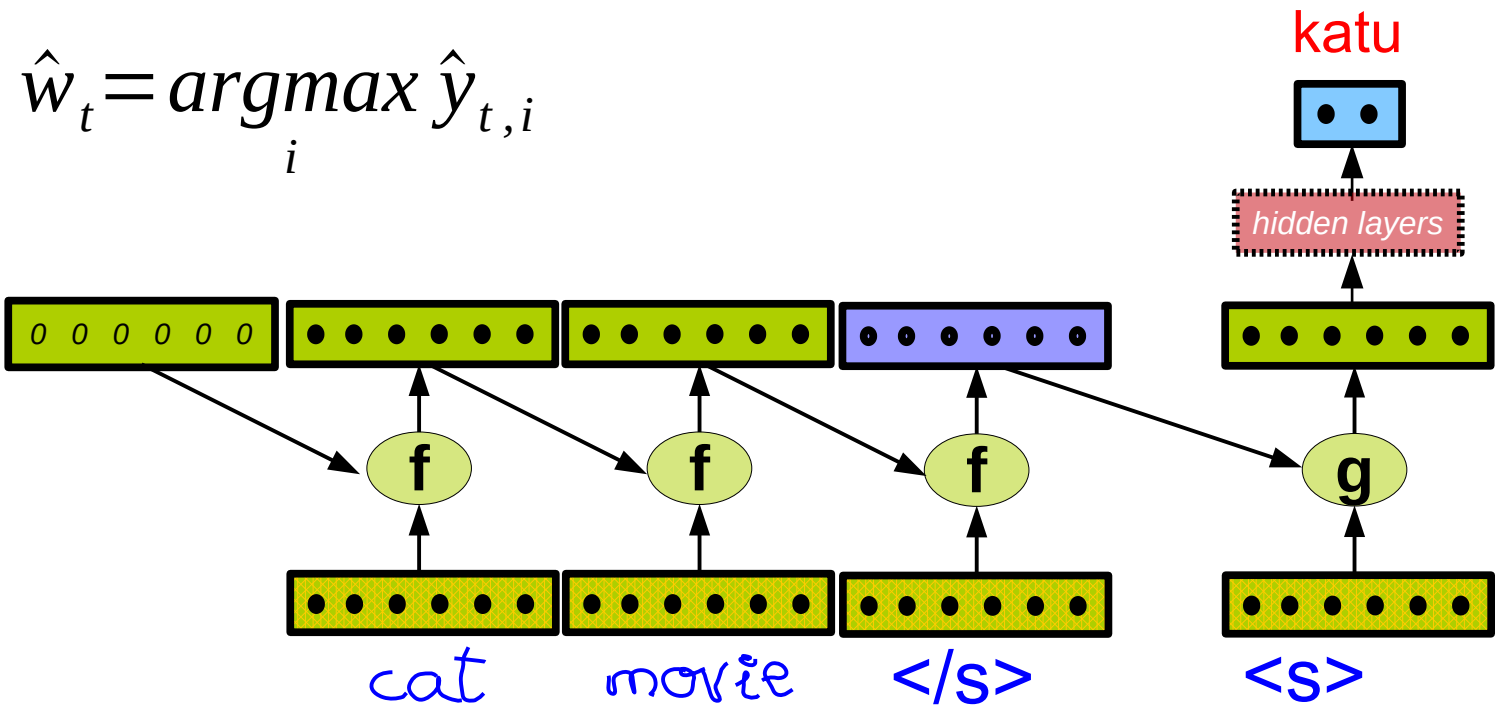
- Test as decoder
  - Compute sentence representation



Sequence to sequence for MT

- Test as conditional RLM decoder
  - Compute sentence representation
  - Generate first word

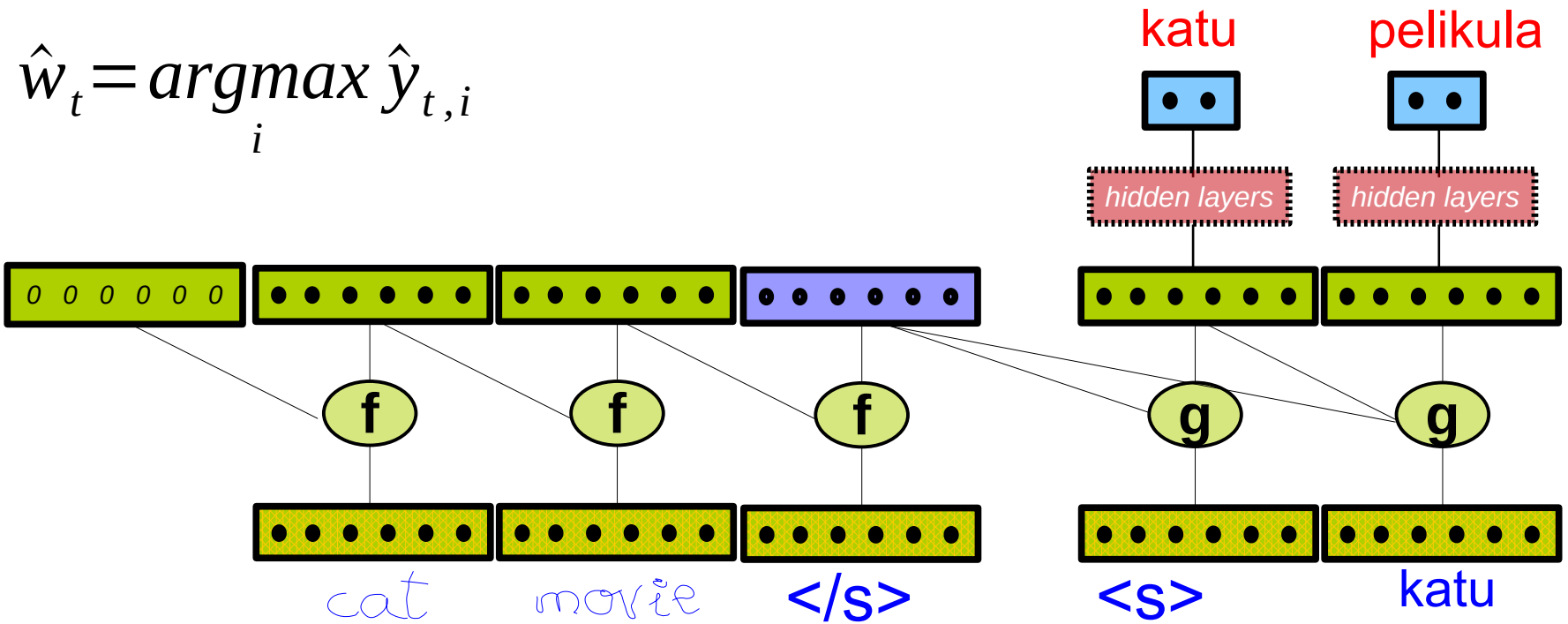
$$\hat{w}_t = \underset{i}{argmax}\, \hat{y}_{t,i}$$



Sequence to sequence for MT

- Test as conditional RLM decoder
  - Compute sentence representation
  - Generate second word

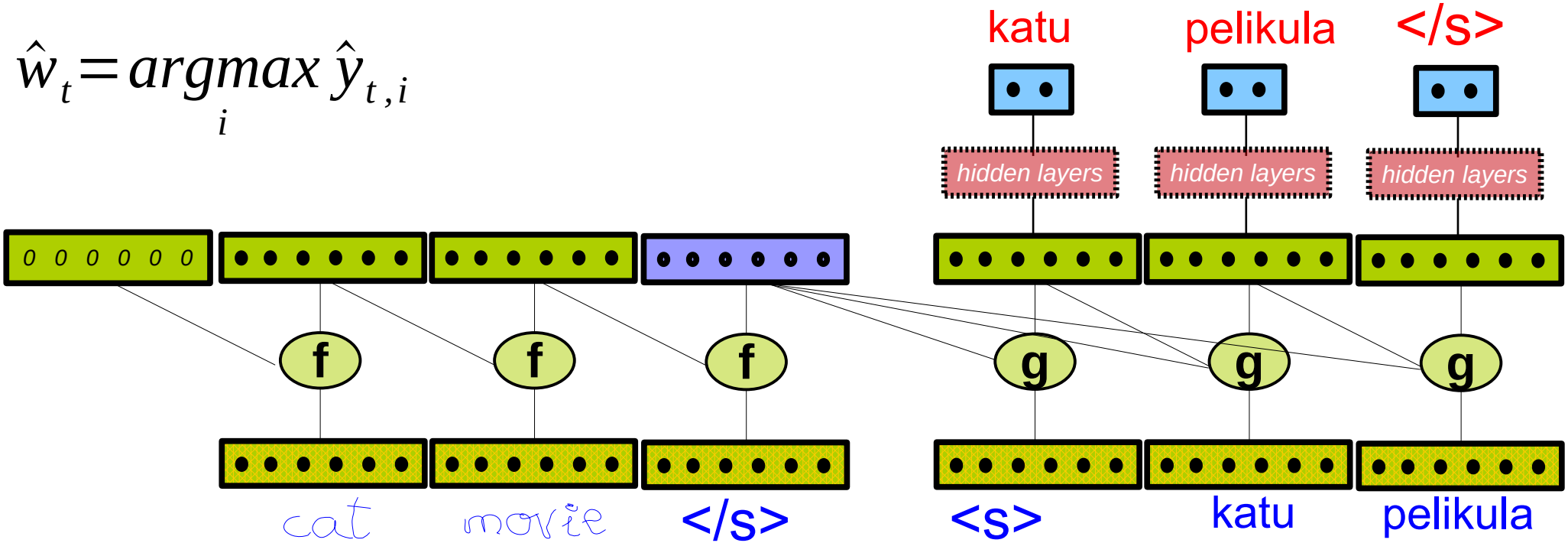
$$\hat{w}_t = \underset{i}{argmax}\, \hat{y}_{t,i}$$



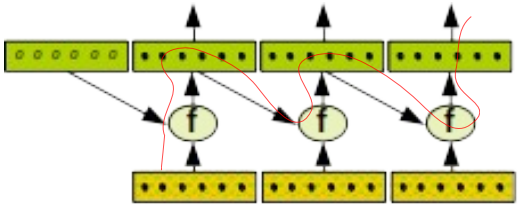
Sequence to sequence for MT

- Test as conditional RLM decoder
  - Compute sentence representation
  - Generate next word, stop if </s>

$$\hat{w}_t = \underset{i}{argmax}\, \hat{y}_{t,i}$$



Training RNNs is hard



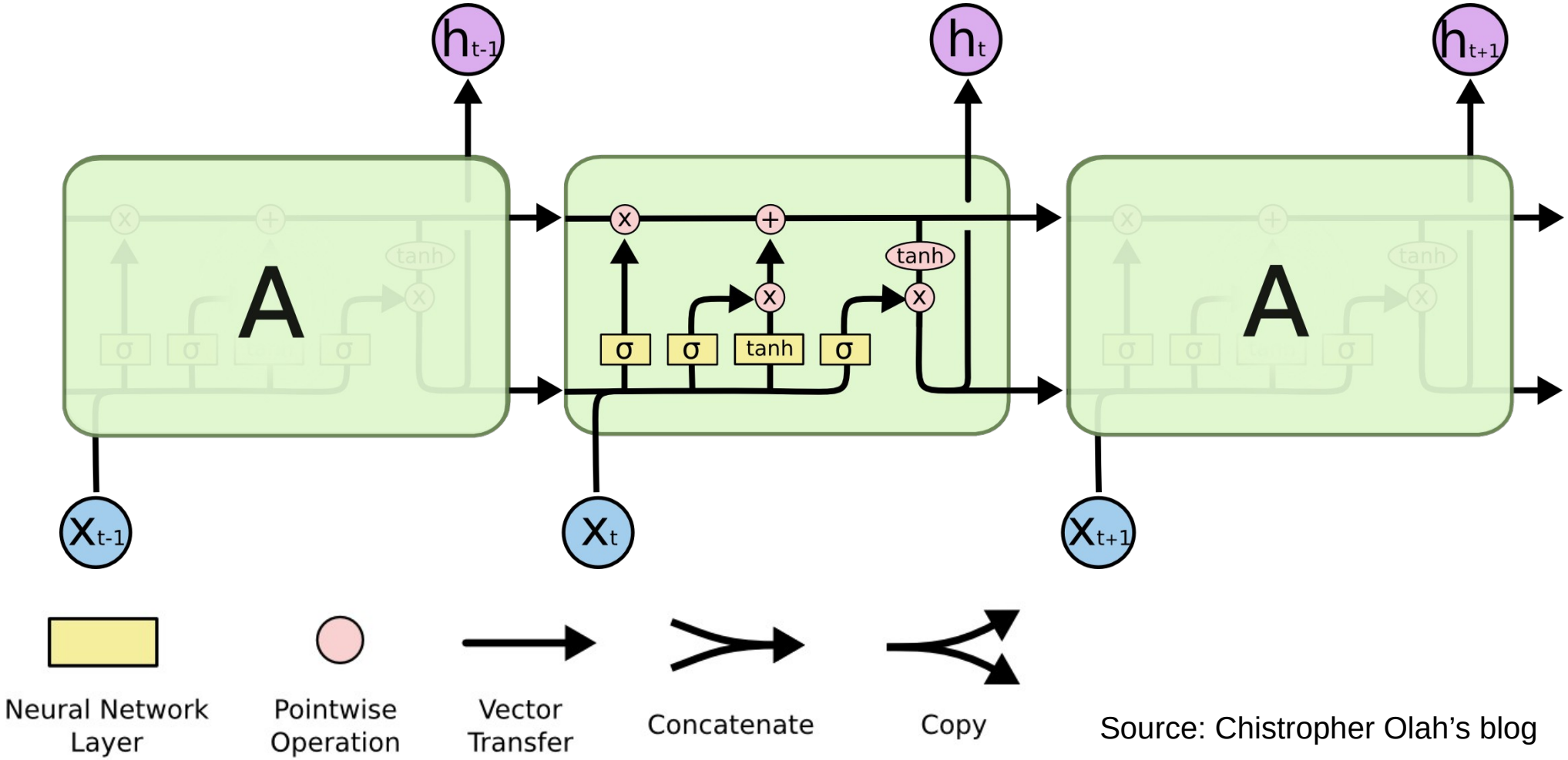
$$\vec{h}_t = \tanh(W[\vec{h}_{t-1}, \vec{w}_t] + \vec{b})$$

(*Basic*) RNNs are unstable, need to carefully initialize parameters

- RNNs are deep:
  layers between first token and last output
  - Actually deep MLPs have similar issues
- Forward: multiply same matrix each time
- Backward: the gradients for early tokens are extremely low

Long Short-Term Memory LSTM

LSTM cells have a rich internal structure

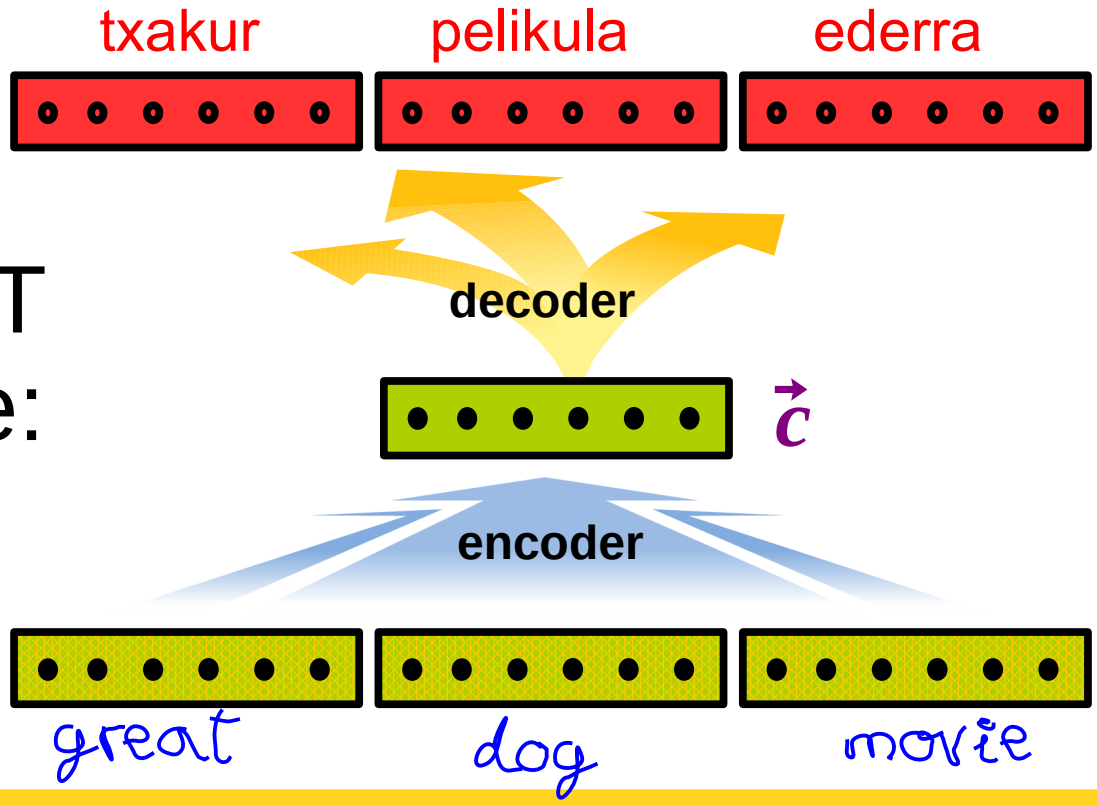


Source: Chistropher Olah's blog

Re-thinking seq2seq

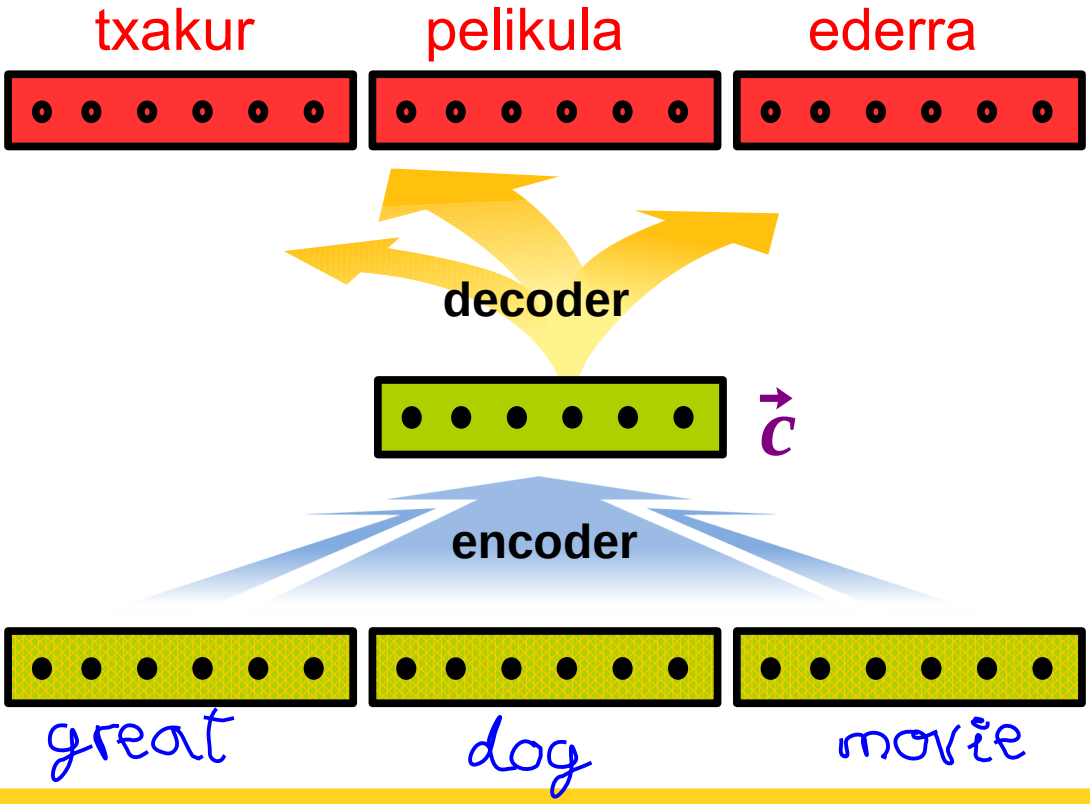
- Encode input sequence into vector $\vec{c}$
- Decode $\vec{c}$ into target sequence

The existing NMT proposal at the time:



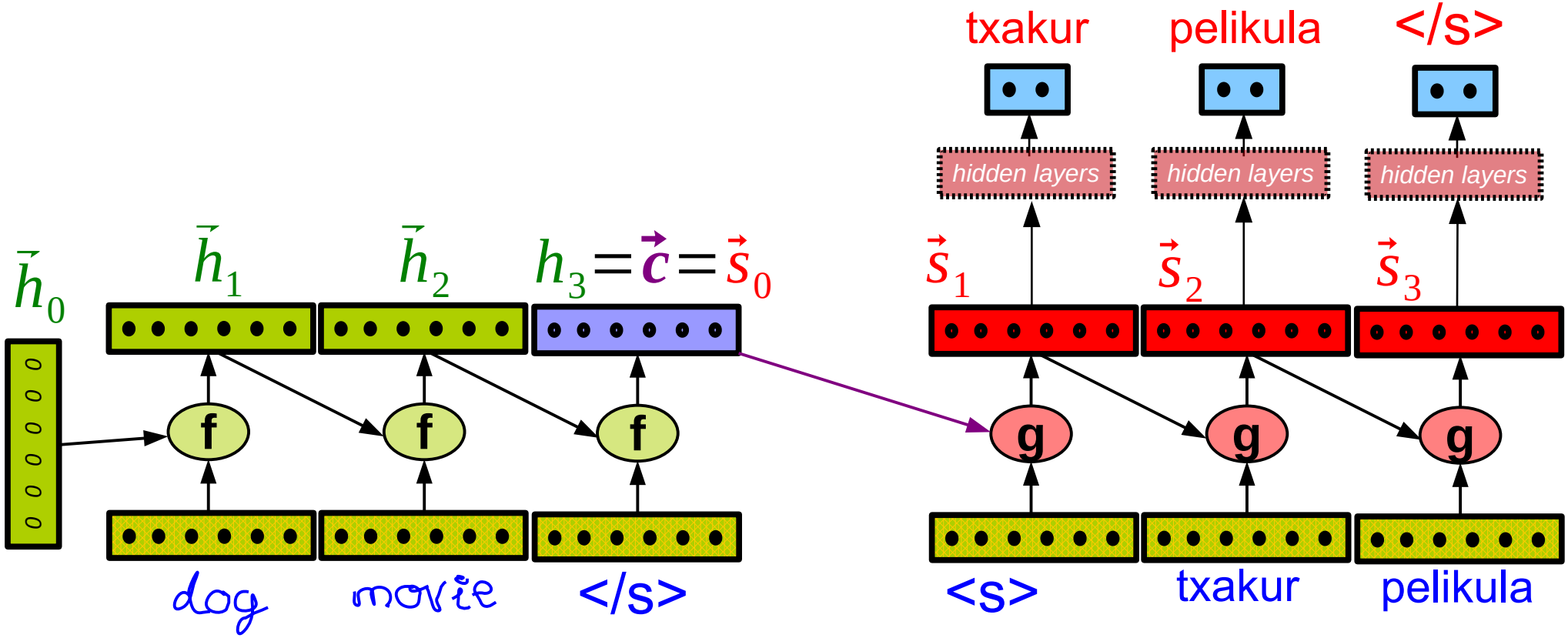
Re-thinking seq2seq

"You can't cram the meaning or a whole %&! $# sentence into a single $&!#* vector!"
(Ray Mooney – Cho's talk at dl4mt workshop 2015)



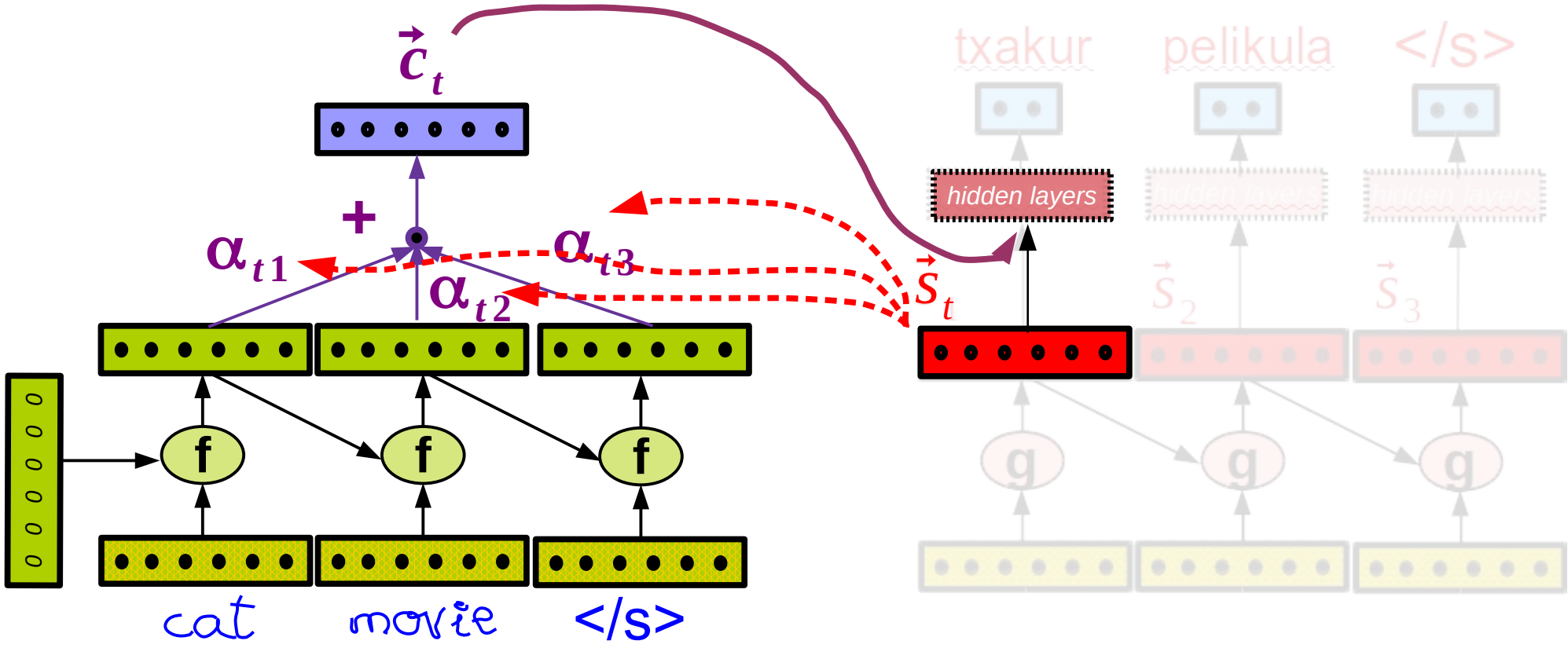
Re-thinking seq2seq for NMT

How can we access the necessary information at each decoding step?



Re-thinking seq2seq for NMT

Attention layer
(Bahdanu et al. 2015; Luong et al. 2015)

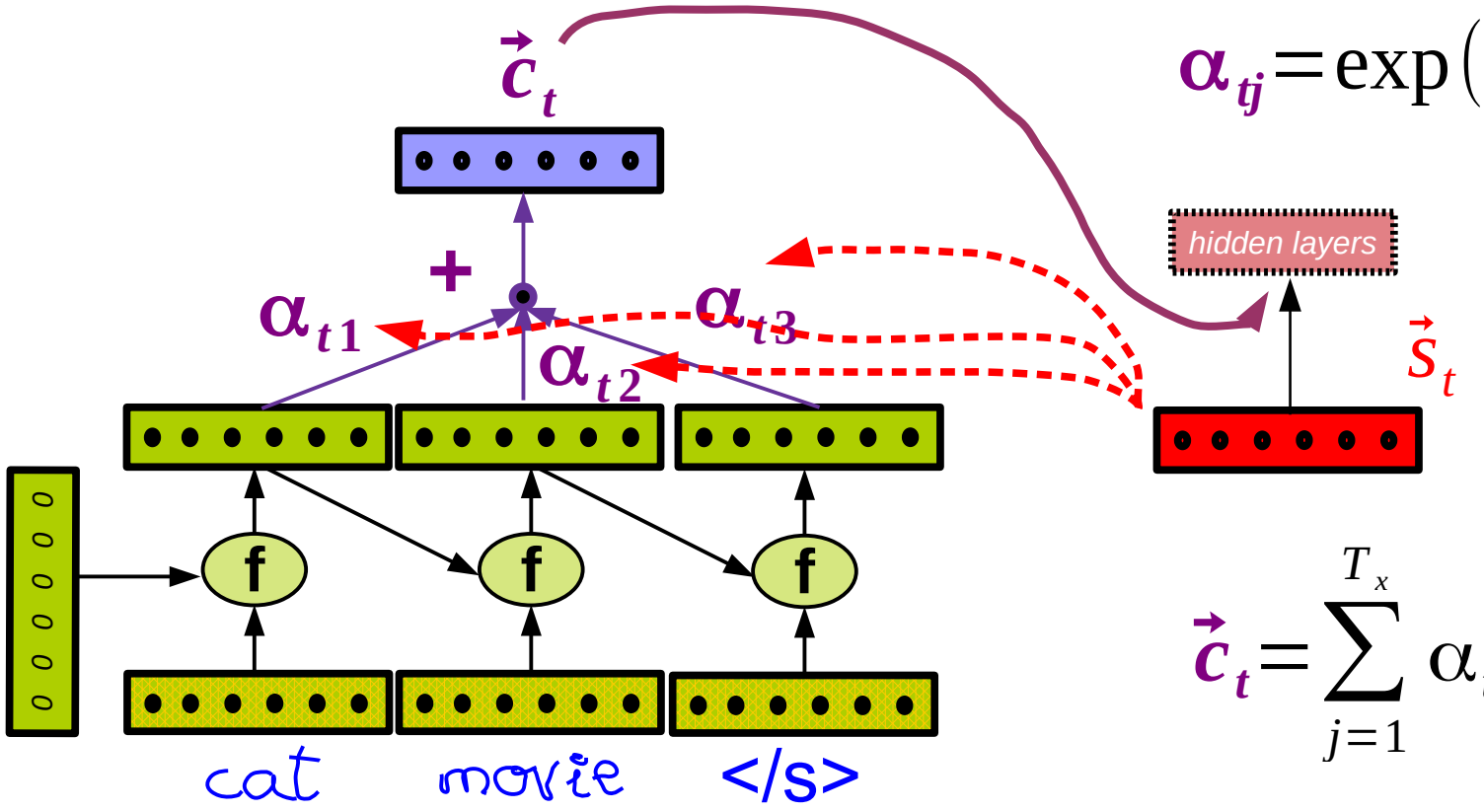


Re-thinking seq2seq for NMT

Attention layer

$$e_{tj} = a(\vec{s}_t, \vec{h}_j) = \vec{s}_t W_a \vec{h}_j$$

$$\alpha_{tj} = \exp(e_{tj}) / \sum_{k=1}^{T_x} \exp(e_{tk})$$

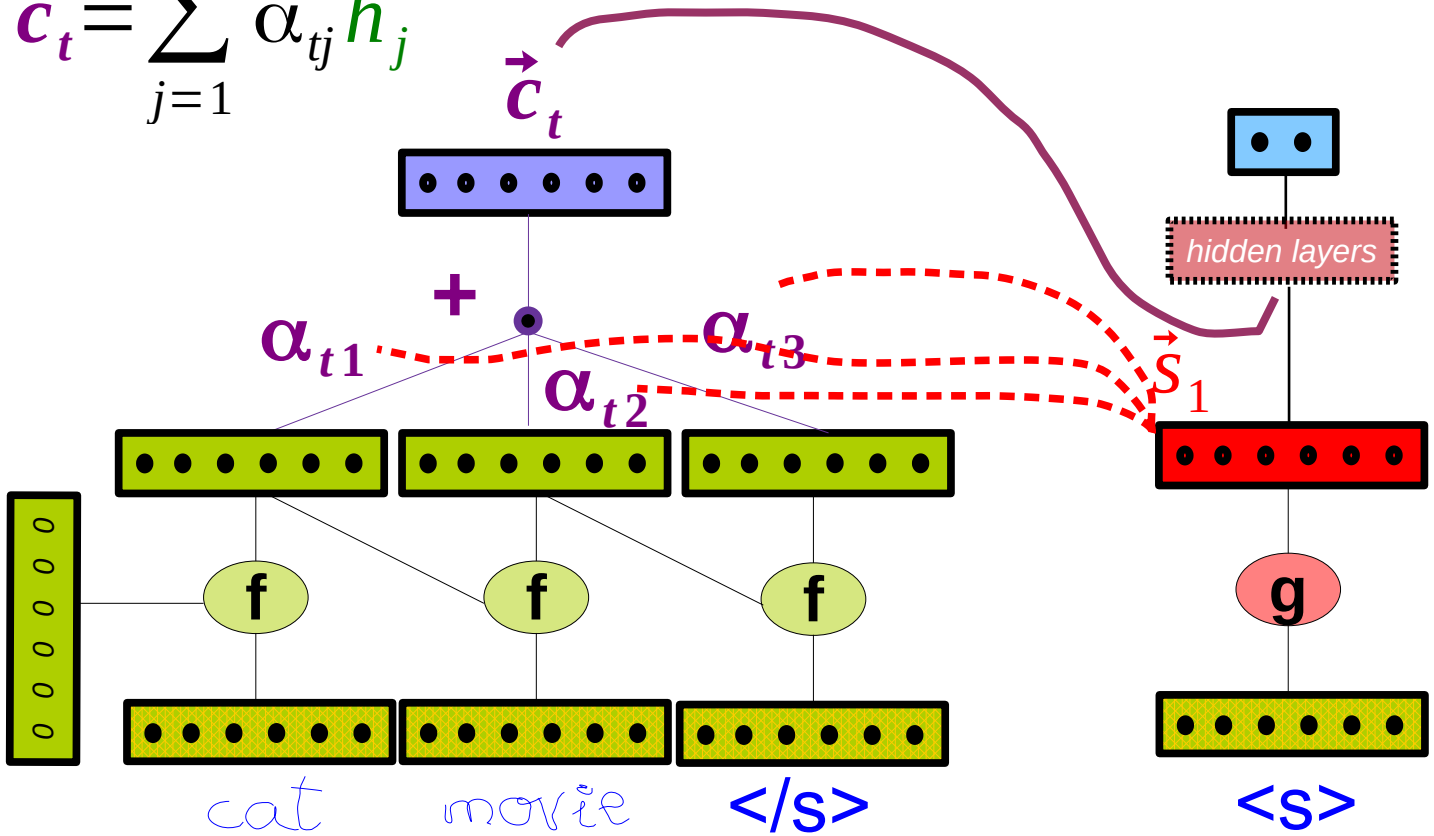


$$\vec{c}_t = \sum_{j=1}^{T_x} \alpha_{tj} \vec{h}_j$$

Re-thinking seq2seq for NMT

$$e_{tj} = a(\vec{s}_t, \vec{h}_j) = \vec{s}_t W_a \vec{h}_j \qquad \alpha_{tj} = \exp(e_{tj}) / \sum_{k=1}^{T_x} \exp(e_{tk})$$

$$\vec{c}_t = \sum_{j=1}^{T_x} \alpha_{tj} \vec{h}_j$$

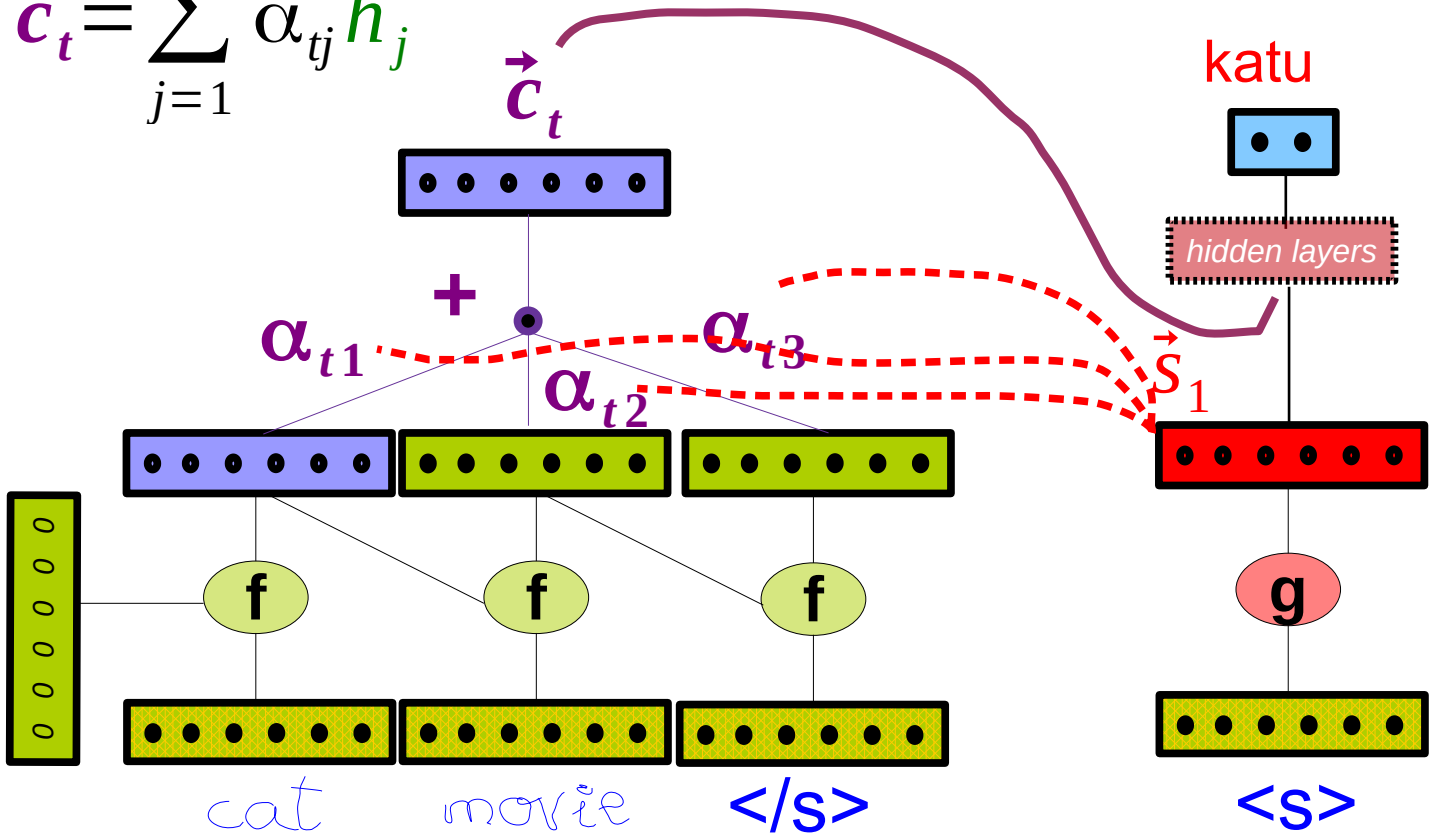


t=1

Re-thinking seq2seq for NMT

$$e_{tj} = a(\vec{s}_t, \vec{h}_j) = \vec{s}_t W_a \vec{h}_j \qquad \alpha_{tj} = \exp(e_{tj}) / \sum_{k=1}^{T_x} \exp(e_{tk})$$

$$\vec{c}_t = \sum_{j=1}^{T_x} \alpha_{tj} \vec{h}_j$$

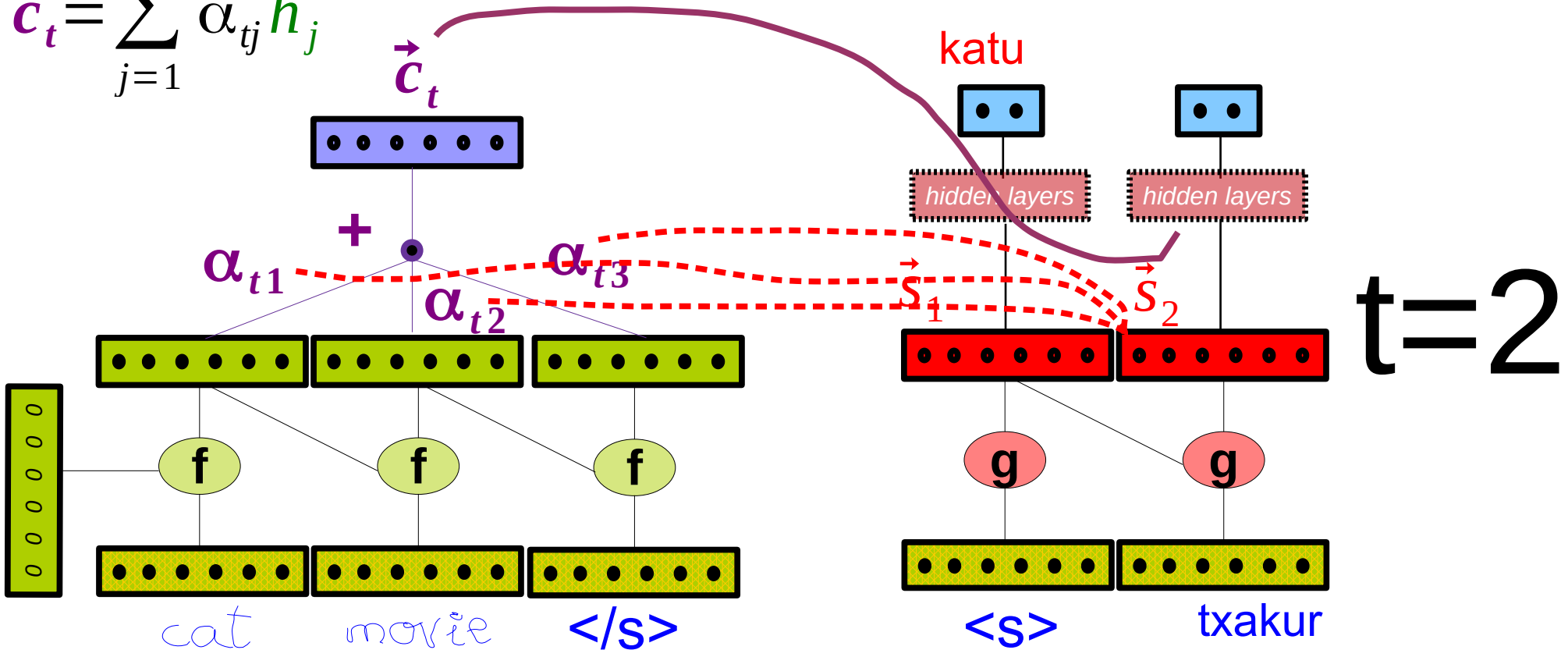


t=1

Re-thinking seq2seq for NMT

$$e_{tj} = a(\vec{s}_t, \vec{h}_j) = \vec{s}_t W_a \vec{h}_j \qquad \alpha_{tj} = \exp(e_{tj}) / \sum_{k=1}^{T_x} \exp(e_{tk})$$

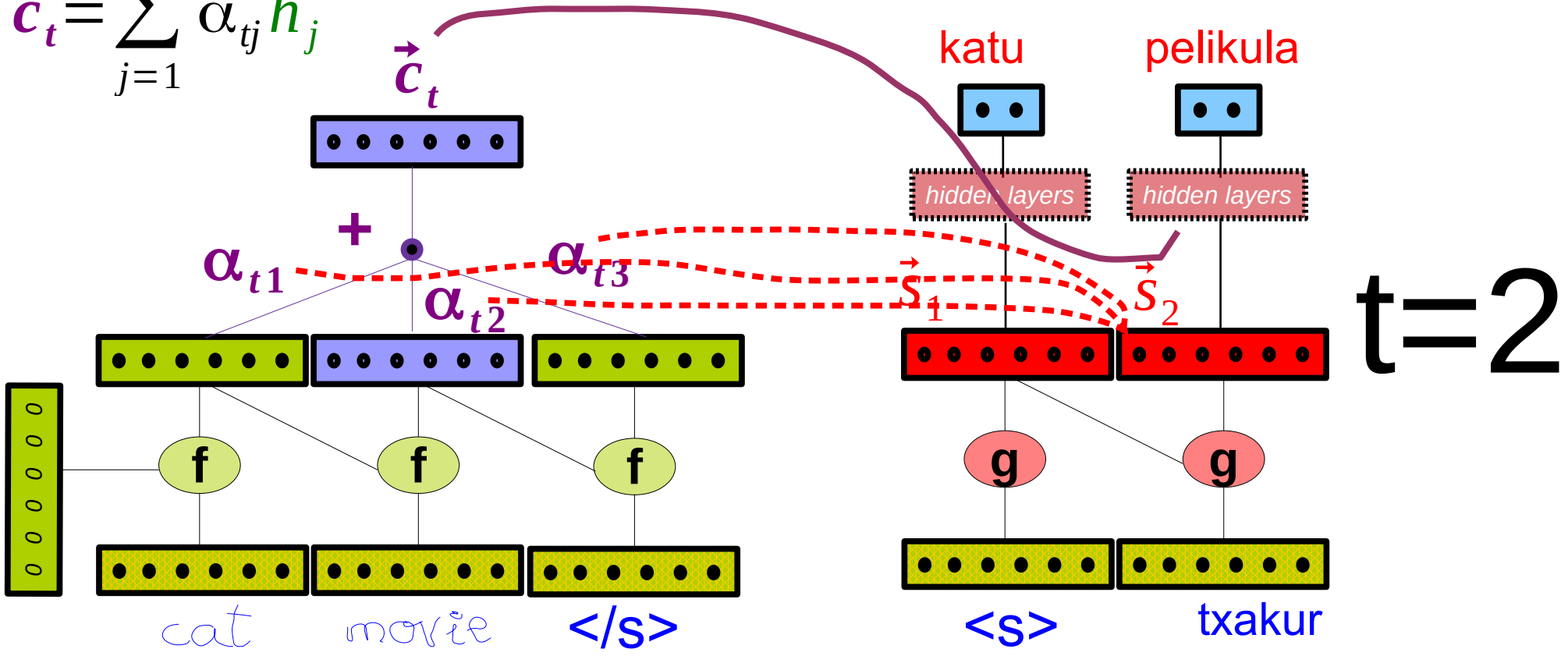
$$\vec{c}_t = \sum_{j=1}^{T_x} \alpha_{tj} \vec{h}_j$$



Re-thinking seq2seq for NMT

$$e_{tj} = a(\vec{s}_t, \vec{h}_j) = \vec{s}_t W_a \vec{h}_j \qquad \alpha_{tj} = \exp(e_{tj}) / \sum_{k=1}^{T_x} \exp(e_{tk})$$

$$\vec{c}_t = \sum_{j=1}^{T_x} \alpha_{tj} \vec{h}_j$$

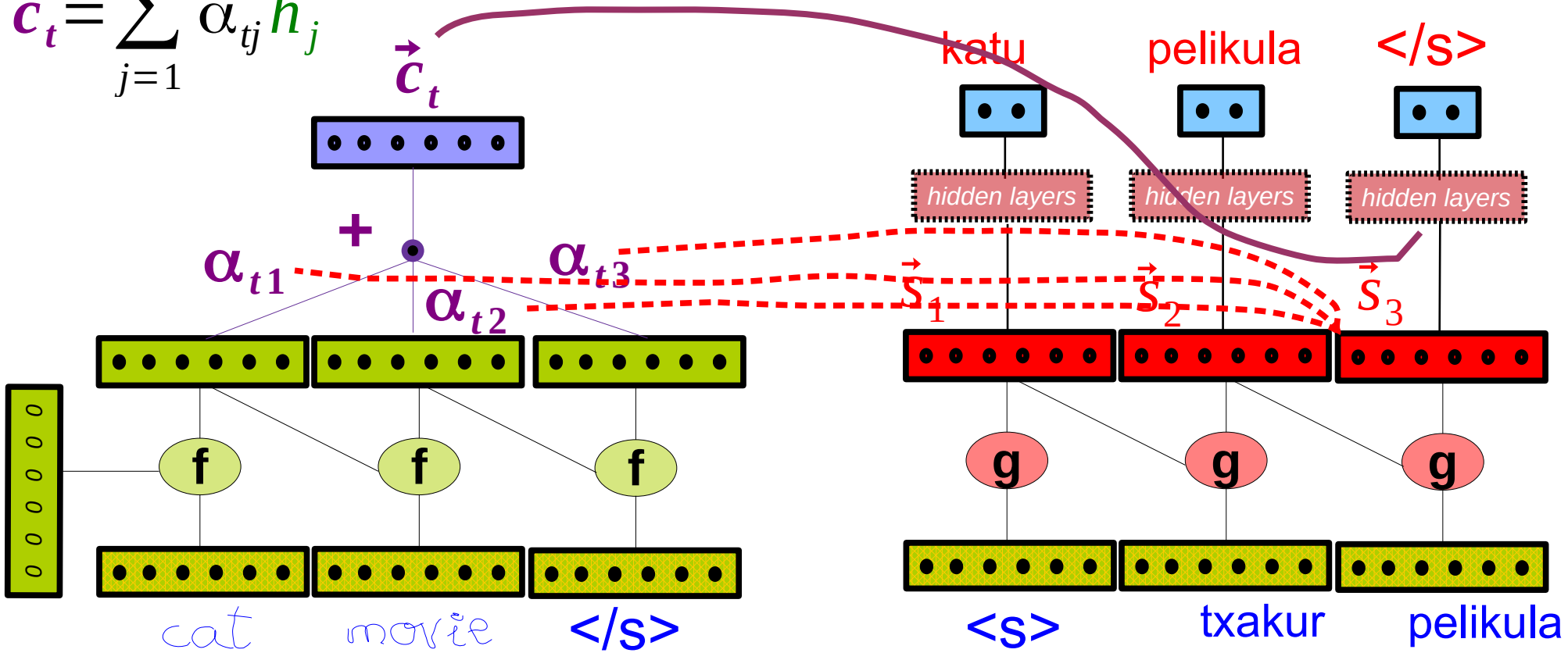


t=2

Re-thinking seq2seq for NMT

$$e_{tj} = a(\vec{s}_t, \vec{h}_j) = \vec{s}_t W_a \vec{h}_j \qquad \alpha_{tj} = \exp(e_{tj}) / \sum_{k=1}^{T_x} \exp(e_{tk})$$

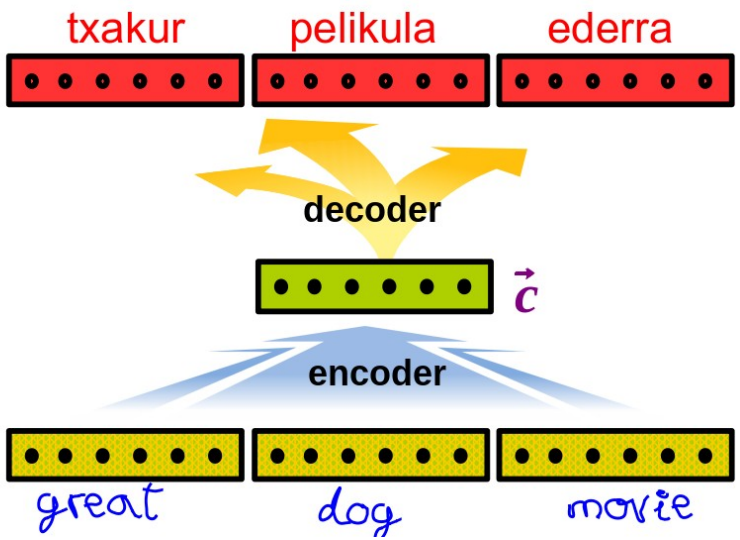
$$\vec{c}_t = \sum_{j=1}^{T_x} \alpha_{tj} \vec{h}_j$$



Re-thinking seq2seq for NMT

Back to sentence representations:

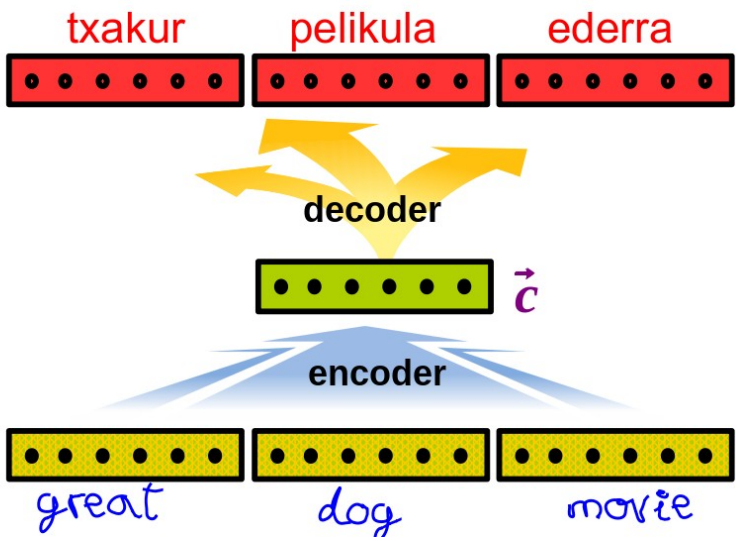
- Instead of last hidden state ...

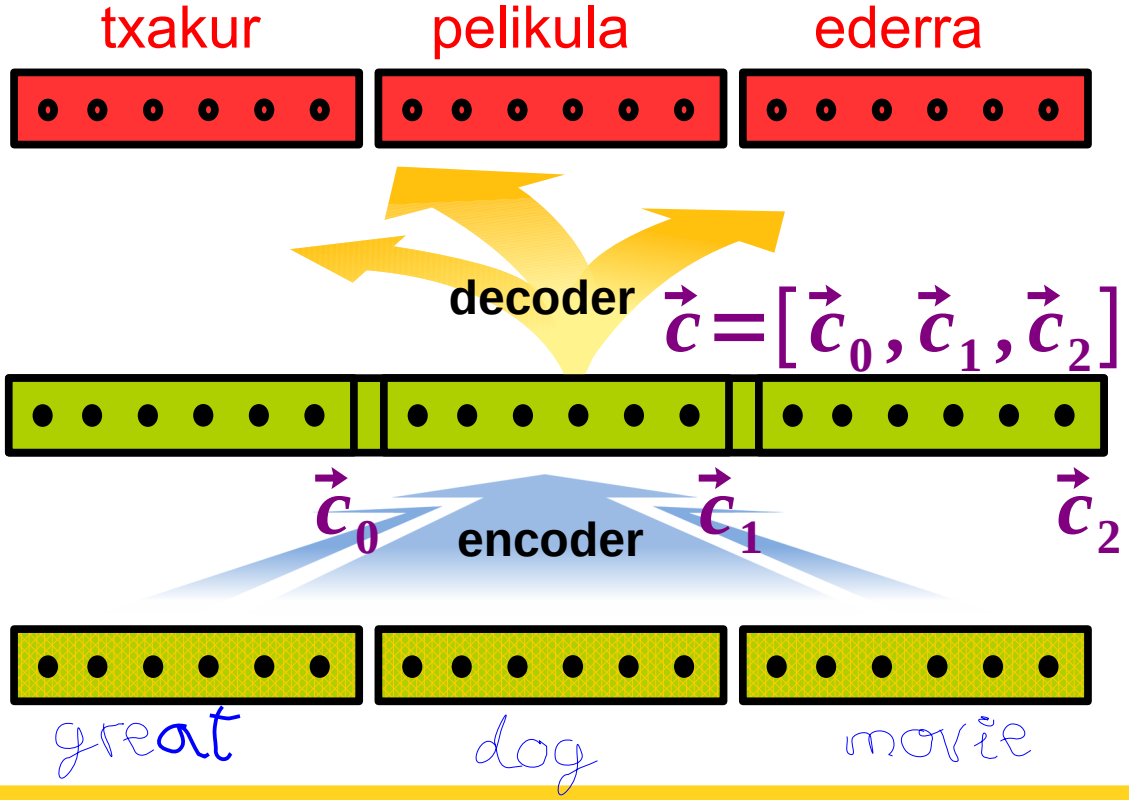


Re-thinking seq2seq for NMT

Back to sentence representations:

- Instead of hidden state ...
- Attention: **all** hidden states





Attention is useful

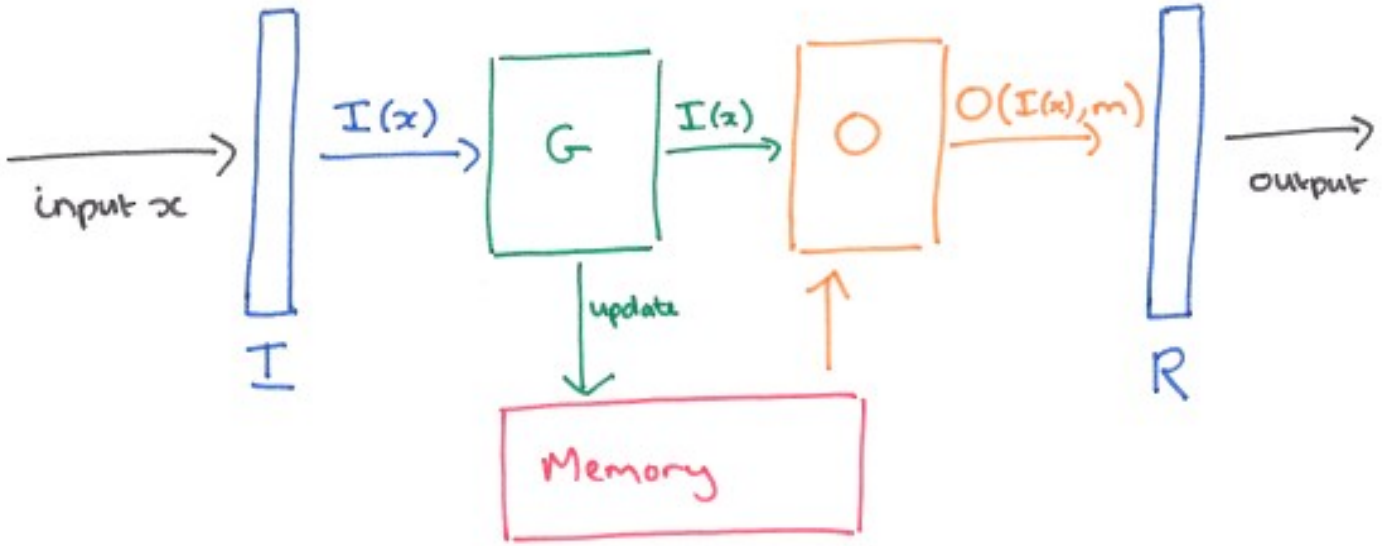
Caption generation:

Source: Xu et al. 2015 - Arxiv 1502.03044

Attention is useful

Memory networks (Sukhbaatar et al. 2016) (also Neural Turing Machine) :

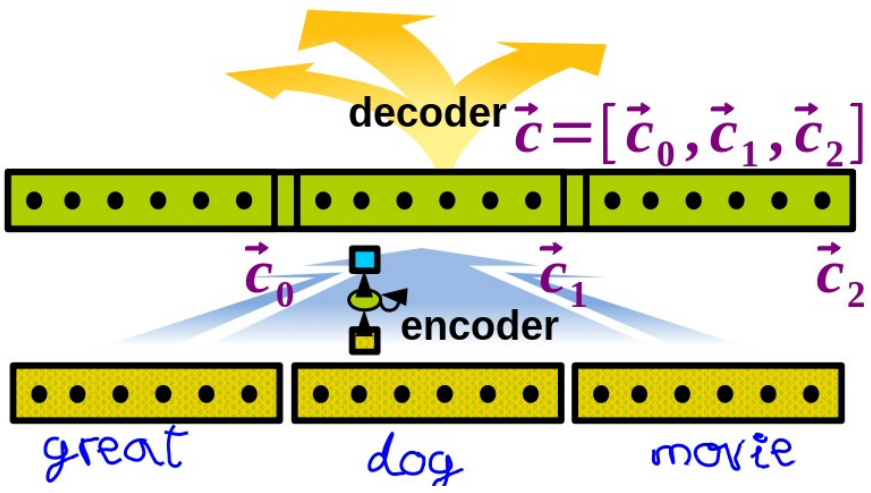
- Set up external knowledge base (memory)
- Use input to select memory cells via attention
- Update selected memory cells

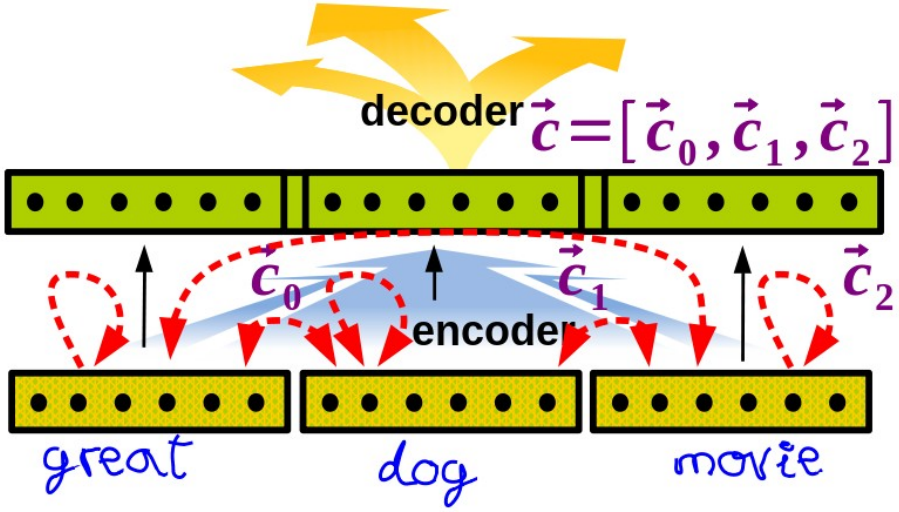


Attention is all you need (!?)

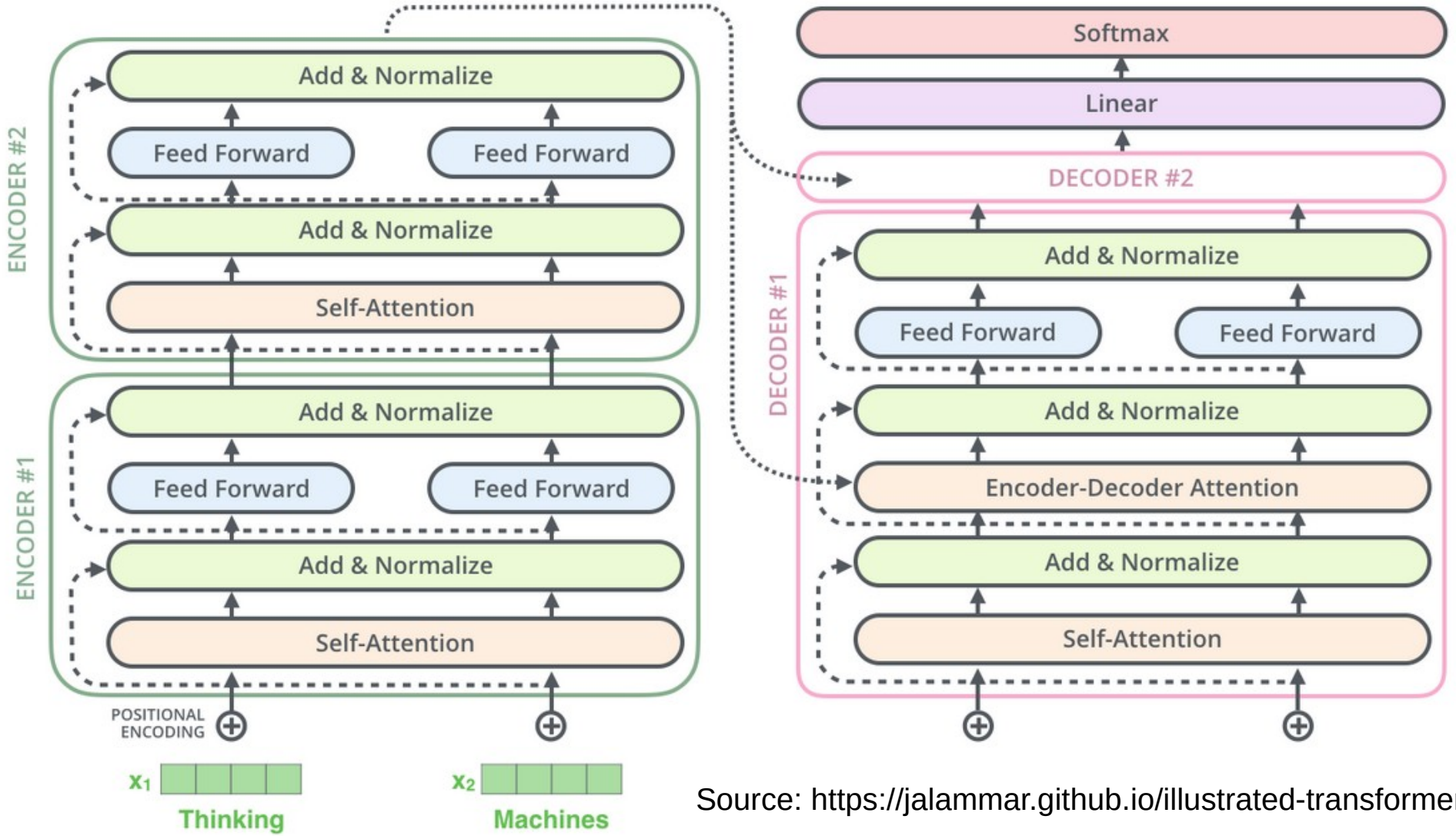
Transformers (Vaswani et al. 2017)

- Extend attention to self-attention: source (and target) with itself
- Build hidden states using feed forward networks





Attention is all you need (!?)



Source: https://jalammar.github.io/illustrated-transformer/

Pre-training transformers

Pre-trained transformers have revolutionized NLP, e.g. GPT-3, BERT

Support The Guardian
Available for everyone, funded by readers
Contribute → Subscribe →
Sign in
The Guardian

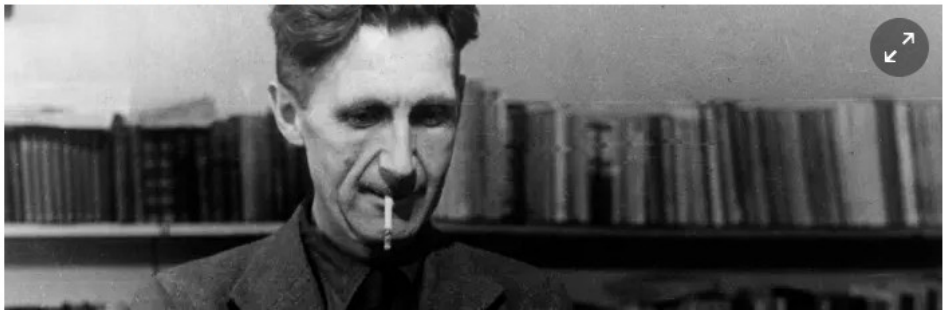
News Opinion Sport Culture Lifestyle

World UK Environment Science Cities Global development Football Tech Business More

Artificial intelligence (AI)

New AI fake text generator may be too dangerous to release, say creators

The Elon Musk-backed nonprofit company OpenAI declines to release research publicly for fear of misuse



Forbes Billionaires Innovation Leadership Money Business Small Business Lifestyl

534 views | Nov 14, 2019, 12:56pm

Google Uses BERT Technology To Develop Its Search Results

Ilker Koksal Contributor
Enterprise & Cloud

Google has recently gone live with their latest update that involves the use of BERT technology in search engine results. According to HubSpot, Google processes over 70 000 search inquiries per second. If you do the math, this number leads to 5.8 billion searches per day.

Besides making such large amount of searches per second possible, Google is also working on improving search results to provide specific and targeted content. The use of BERT and the new neural networking techniques will lead to significant changes in the search algorithm.

Pre-training transformers

- Many successful pretraining approaches are based on language modeling
- Informally, a LM learns some variant of

  $P_\theta(\textit{text})$ or $P_\theta(\textit{text} \mid \textit{some other text})$

- Doesn't require human annotation
- Many languages have enough text to learn high capacity model
- Versatile—can learn both sentence and word representations with a variety of objective functions

Pre-training transformers

Word embedding methods (e.g. word2vec) learn one vector per word:

cat = [0.1, -0.2, 0.4, ...]

dog = [0.2, -0.1, 0.7, ...]

Pre-training transformers

Word vectors compress all contexts into a *single vector*

Nearest neighbor GloVe vectors to "**play**"

VERB	NOUN	ADJ
playing	game	multiplayer
played	games	
	players	
	football	

Pre-training transformers

Instead of learning one vector per word, learn a vector that depends on context

f(play | The kids play a game in the park.)

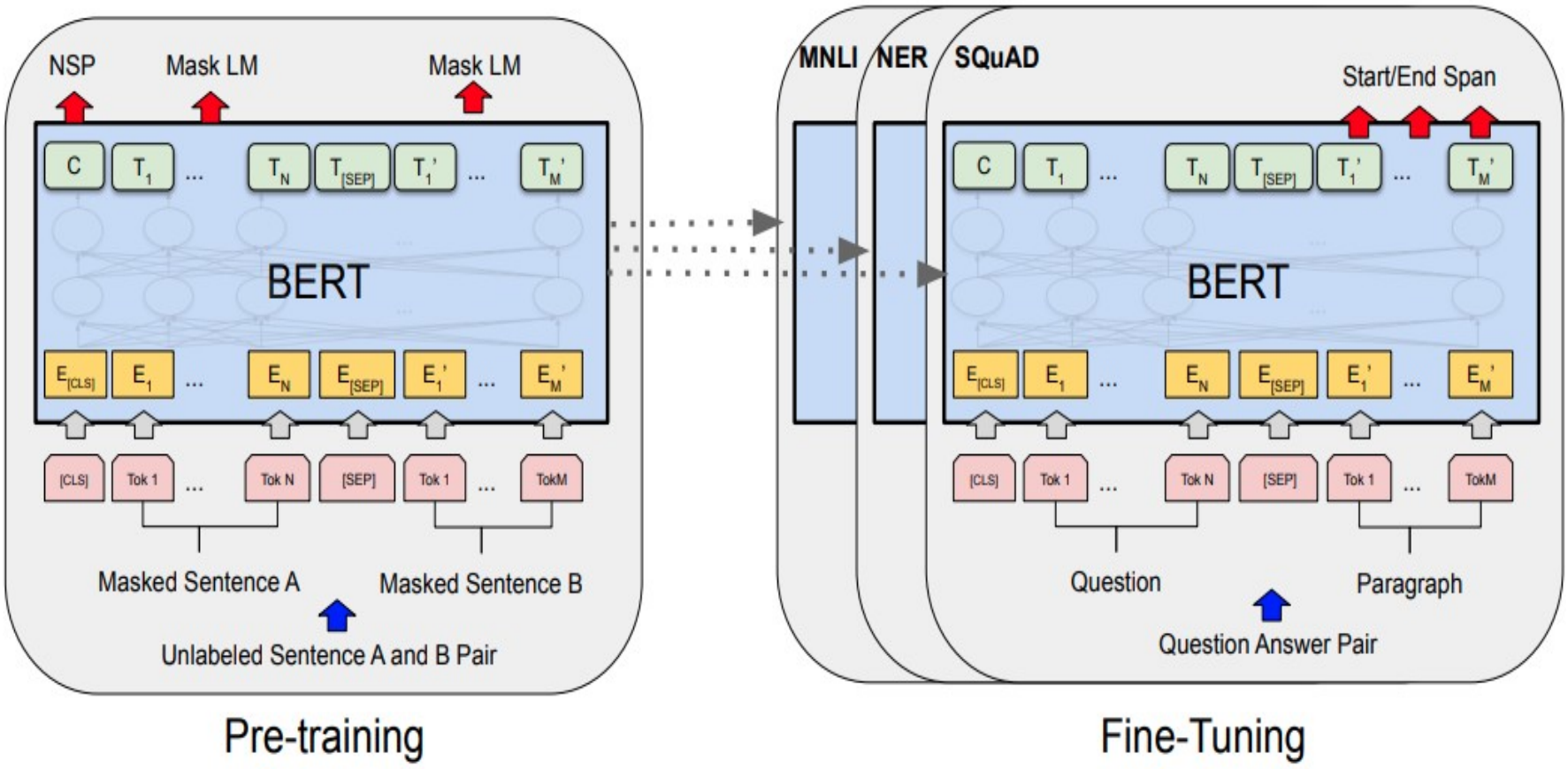
!=

f(play | The Broadway play premiered yesterday.)

Learn **contextual word embeddings**!

Pre-training transformers

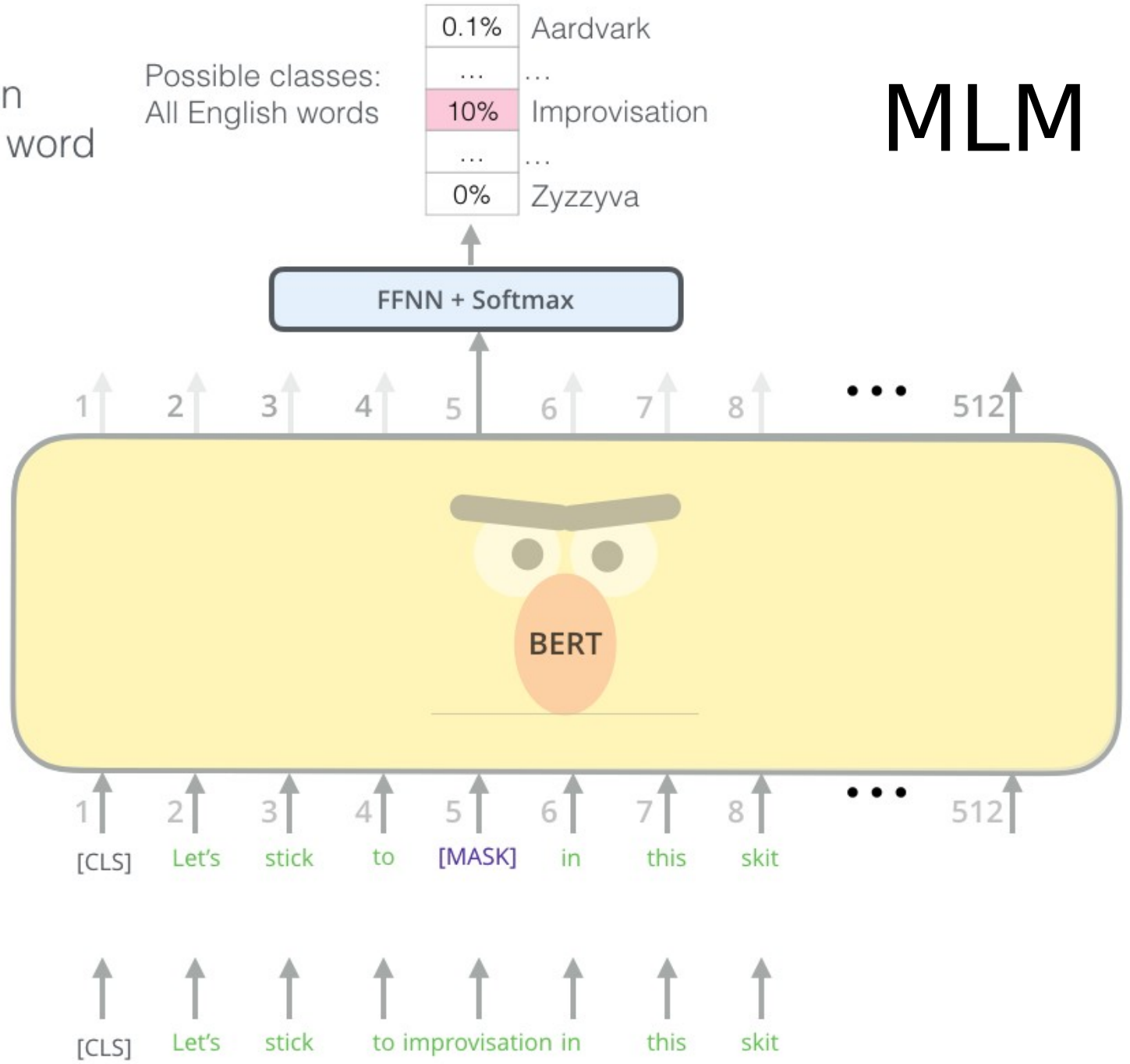
BERT pretrains both sentence and contextual word representations.
Uses masked LM (MLM) and next sentence prediction.
BERT-large has 340M parameters, 24 layers!



(Devlin et al. 2019)

Pre-training transformers

Use the output of the masked word's position to predict the masked word



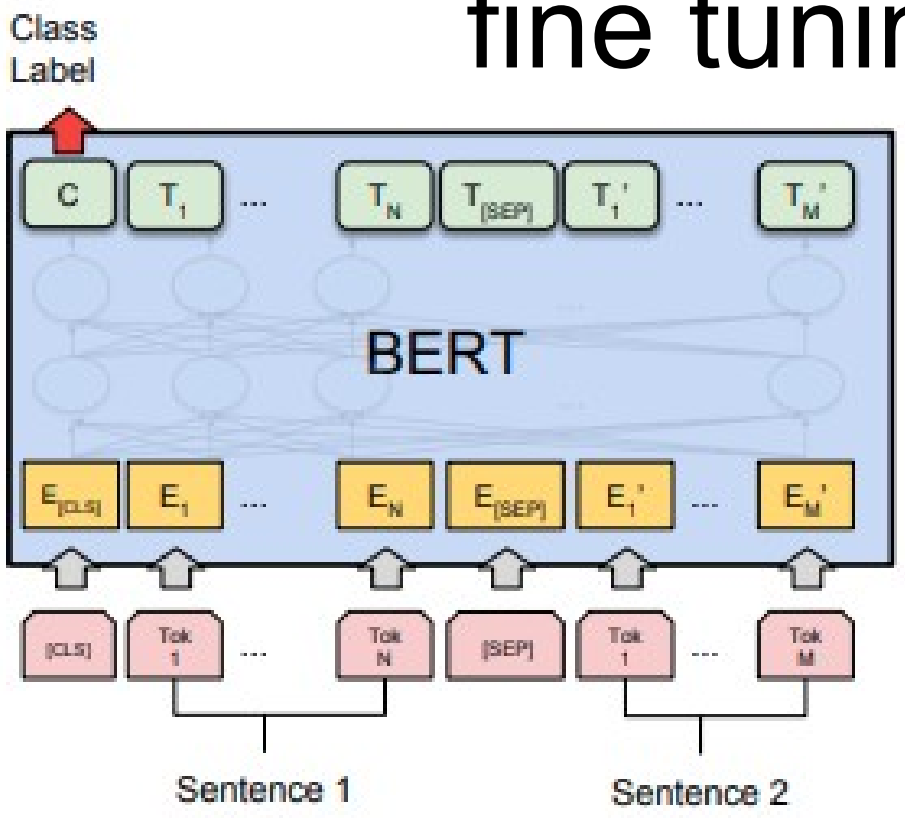
MLM

Randomly mask 15% of tokens

Input

Pre-training transformers

fine tuning on tasks



(a) Sentence Pair Classification Tasks: MNLI, QQP, QNLI, STS-B, MRPC, RTE, SWAG

(Devlin et al. 2019)

Pre-training transformers

Pre-trained transformers have revolutionized NLP and vision, **now without CNNs**

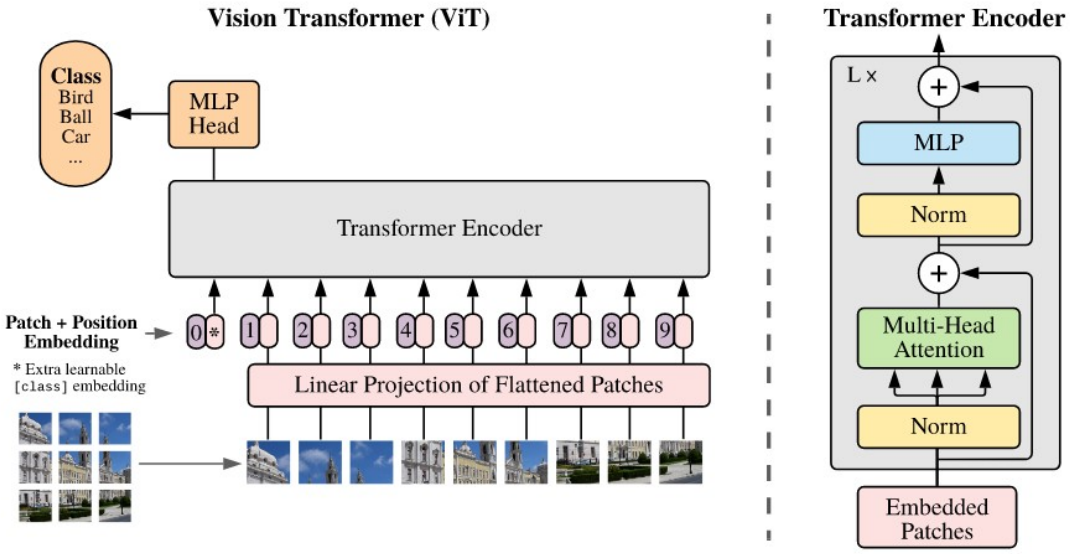


Figure 1: Model overview. We split an image into fixed-size patches, linearly embed each of them, add position embeddings, and feed the resulting sequence of vectors to a standard Transformer encoder. In order to perform classification, we use the standard approach of adding an extra learnable "classification token" to the sequence. The illustration of the Transformer encoder was inspired by Vaswani et al. (2017).

Source: https://arxiv.org/abs/2010.11929

Cross-linguality and machine translation without bilingual data

Eneko Agirre
@eagirre

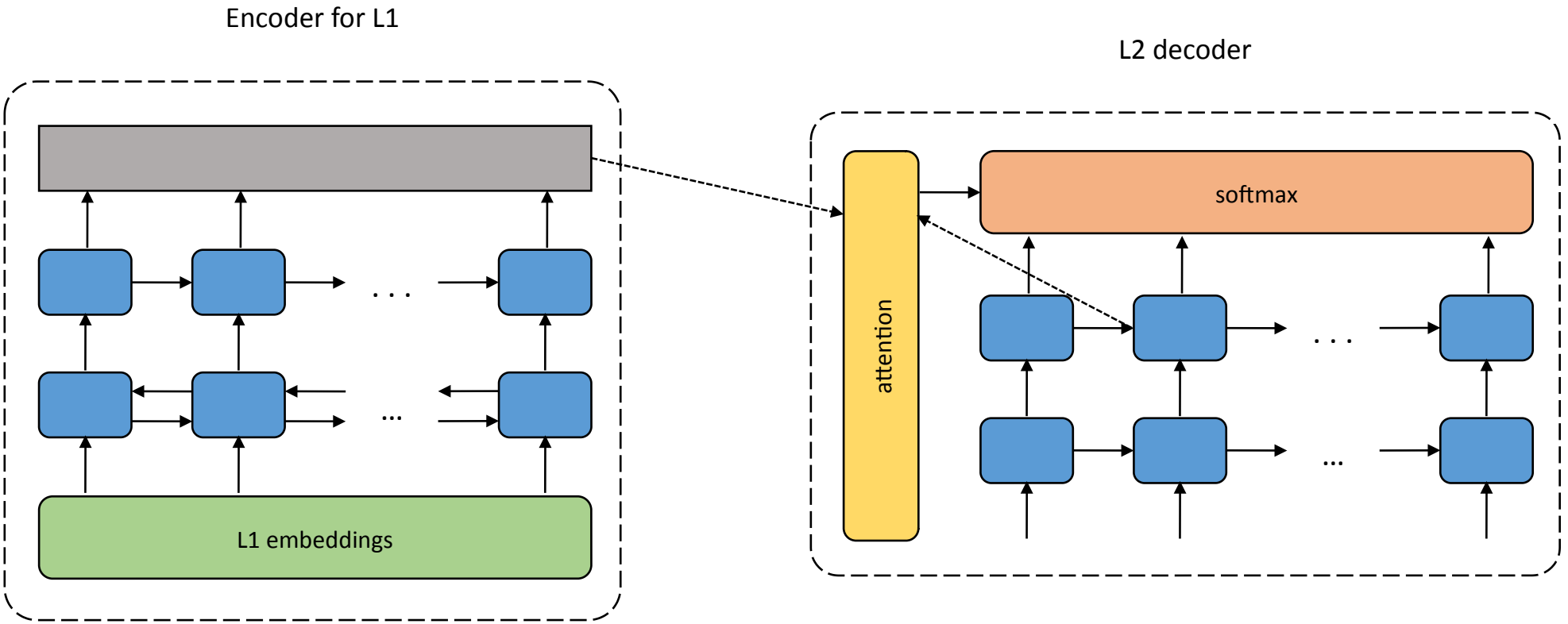
Joint work with: Mikel Artetxe, Gorka Labaka


HiTZ NLP research center - http://hitz.eus
University of the Basque Country (UPV/EHU)

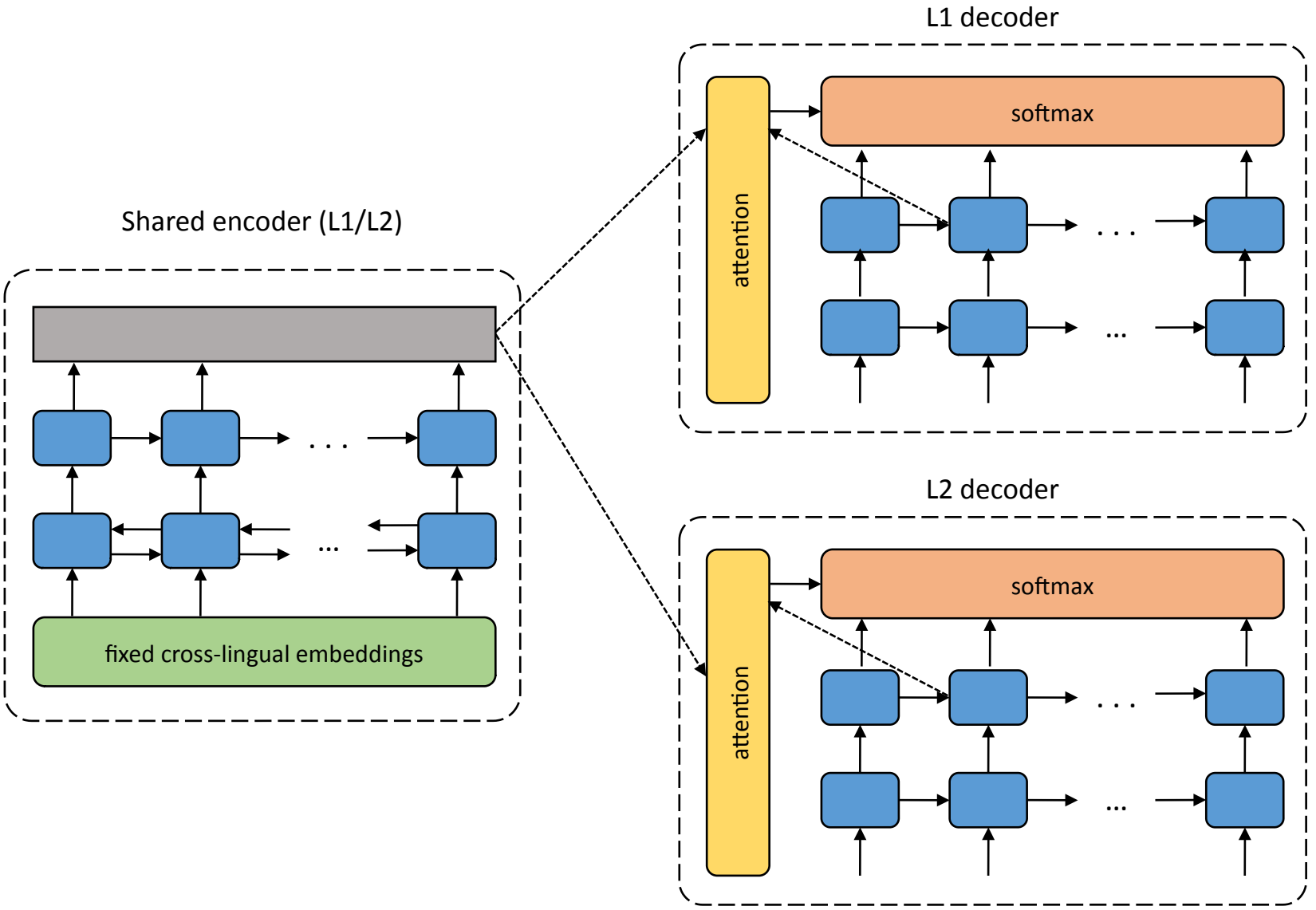
Unsupervised machine translation

- Given:
  - Some books in Chinese
  - Some **other** books in Arabic
  - A person who does not know Chinese nor Arabic
- Can a person learn to translate from Chinese to Arabic or vice versa?
- A program developed here can!

Unsupervised machine translation



Unsupervised machine translation



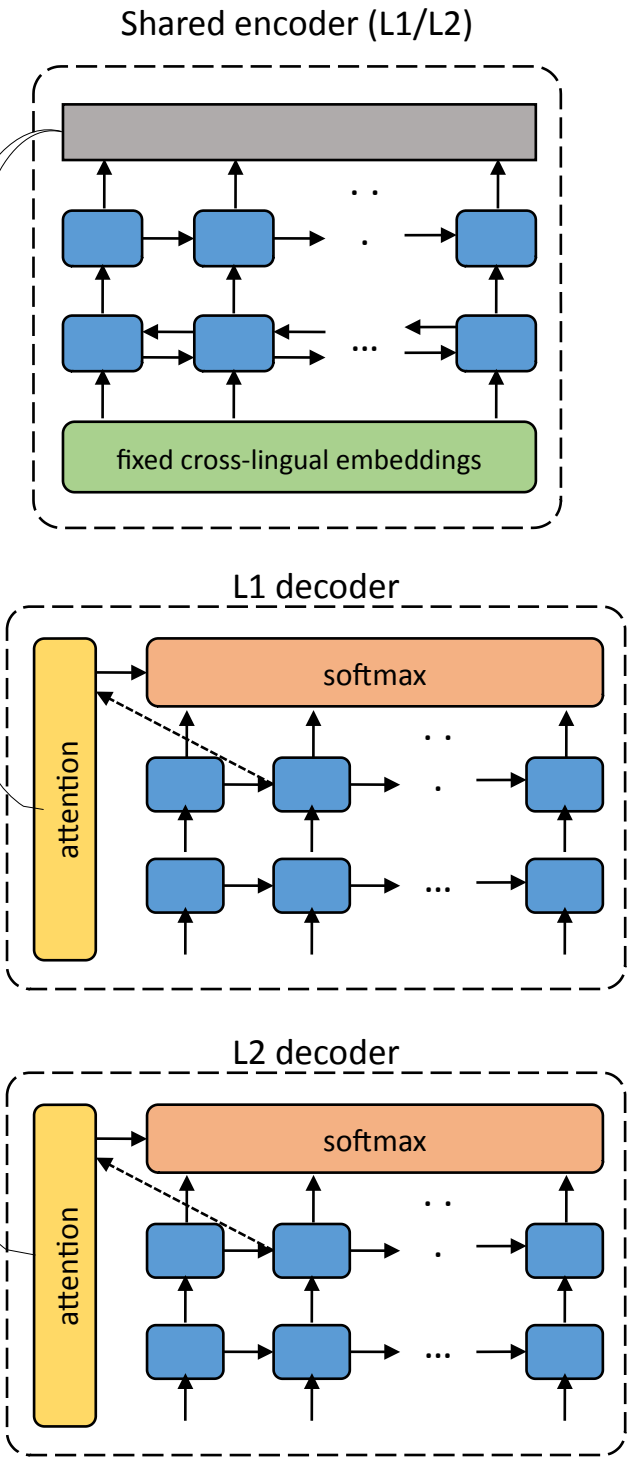
Unsupervised MT

Previously: cross-lingual embeddings

Steps:

- Train as autoencoder:
  - *Given sentence in L1*
    train shared encoder with L1 decoder
  - *Given sentence in L2*
    train shared encoder with L2 decoder
- Test:
  - *Given sentence in L1 produce sentence in L2*
  - *Given sentence in L2 produce sentence in L1*

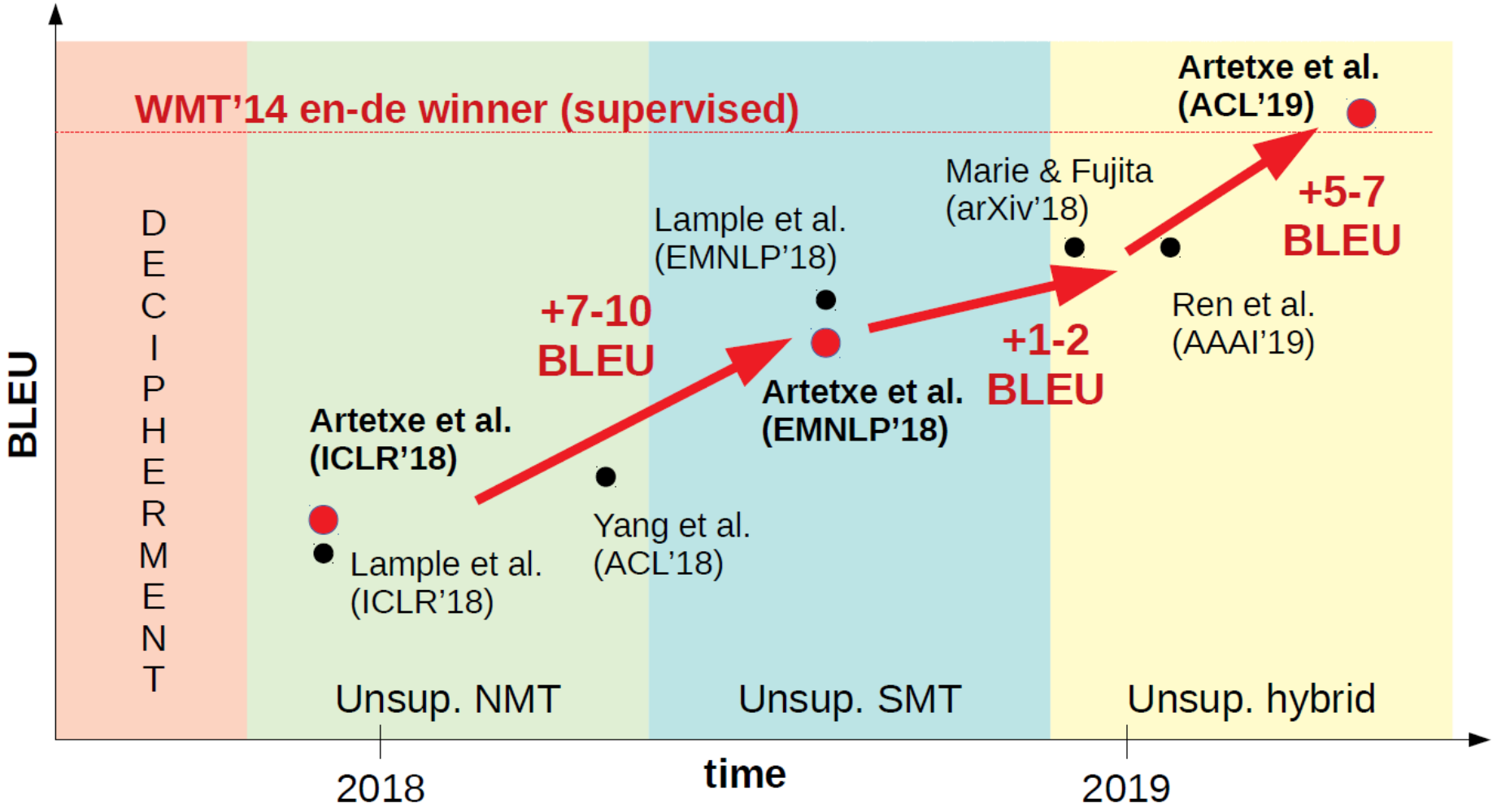
Fails!! Why?



Unsupervised machine translation

- Improvements
  - Denoising autoencoder: introduce noise in input
  - Backtranslation:
    - Given sentence in L1 translate to L2 and train shared encoder with L1 decoder to recover original sentence
- It works!
  - Artetxe et al. (2017): The first paper (by 1 day!) showing that it is possible to translate from one language to the other without bilingual resources
  - Further improvements (Artetxe et al. 2019): performance comparable to the state of the art in 2014 (with no bilingual data!)

Unsupervised machine translation



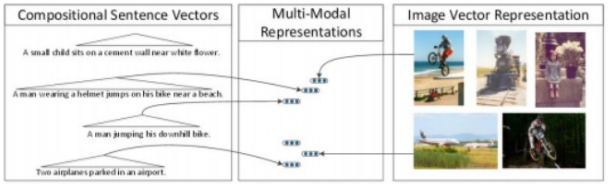
UMT is at the level MT was at 2014

Final words

- We only scratched the surface of representation learning for NLP
- Language understanding
  - Complex structure beyond sequence
  - Function words beyond embeddings (every,not)
- We'll see this in more detail in "Hizkuntzaren Prozesamendua"
- If interested (e.agirre@ehu.eus)
  - Two MLNN project (Oier)
  - Post-graduate course on January: DL4NLP in 14 days (full!) http://ixa2.si.ehu.eus/deep_learning_seminar/
  - Summer course on July DL4NLP in 3 days http://ixa2.si.ehu.eus/deep_learning_seminar/
  - GrAL / TFG with selected students

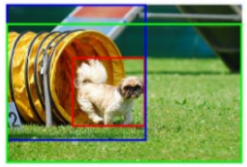
- Master on Language Technologies
  - Erasmus Mundus program https://lct-master.org/
    Very good grants, deadline 1st of February (!)
  - Local program http://ixa.eus/master/
- If interested (e.agirre@ehu.eus)

Multimodal Text Grounding



- Ground text to real world objects and entities
- Extract (space) relation of objects and entities
- Deep-learning for multimodal (text/images) representations
- Semantics of texts

- **Profile**: Computer Scientist
- **Team**: Oier, Aitor, Eneko



A little brown and white dog emerges from a yellow collapsable toy tunnel onto the lawn.

Interactive Q&A in forums

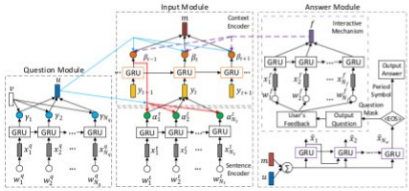
People and machines have knowledge gaps

- When answering a question we often need more background
- Solution: **Interactive Q&A**

The master bedroom is east of the garden.
The guest bedroom is east of the office.
The guest bedroom is west of the hallway.
The bathroom is east of the master bedroom.
Q: What is the bedroom east of?
SQ: Which bedroom, master one or guest one?
FB: Master bedroom
A: Garden

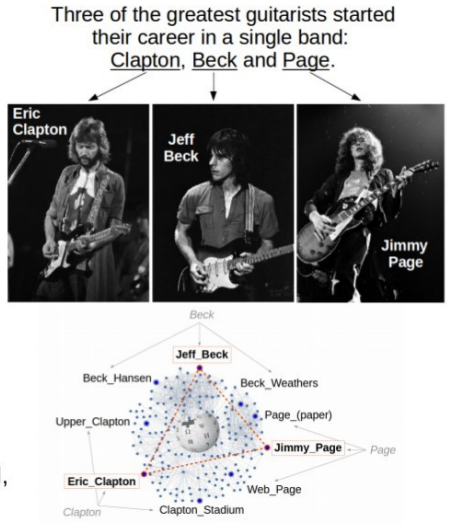
Techniques: Deep Reinforcement Learning, Dialogue
Scenarios: Stack Exchange,
Quora (Kaggle), education

- **Profile**: Computer Scientist
- **Team**: Iñigo, Eneko, Montse Oier, Aitor

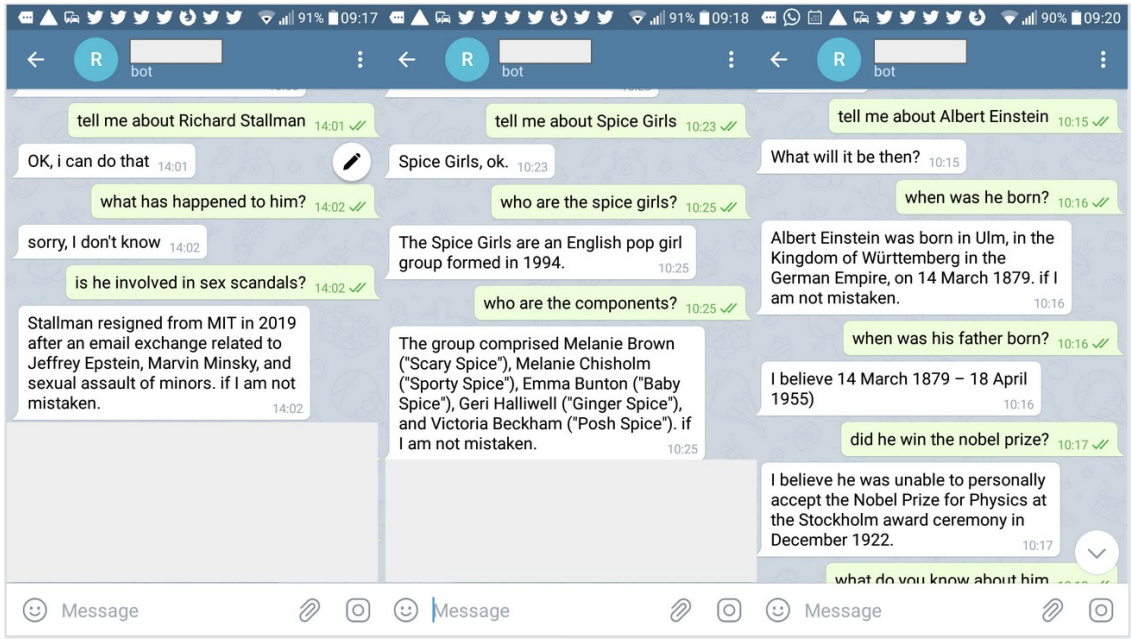


Disambiguation of words and entities

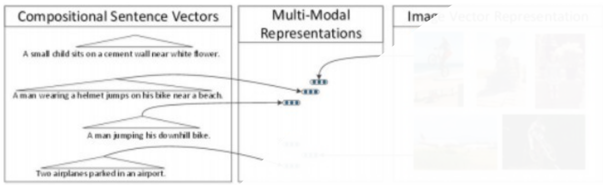
- Linking text to entities in knowledge bases
- Deep learning and graph-based approaches
- Integrate many Kbs: Wikipedia, WordNet, etc.
- Multilingual



Profile: Computer scientist
Team: Aitor, Eneko, Ander, Oier, Josu, Mikel, Gorka.



Multimodal Text Grounding



- Ground text to real world objects and entities
- Extract (space) relation of objects and entities
- Deep-learning for multimodal (text/images) representations
- Semantics of texts

- Profile: Computer Scientist
- Team: Oier, Aitor, Eneko

Interactive Q&A in forums

People and machines have knowledge gaps
- When answering a question we often need more background
- Solution: Interactive Q&A

The master bedroom is east of the garden.
The guest bedroom is east of the office.
The guest bedroom is west of the hallway.
The bathroom is east of the master bedroom.
Q: What is the bedroom east of?
SQ: Which bedroom, master one or guest one?
FB: Master bedroom
A: Garden

Techniques: Deep Reinforcement Learning, Dialogue
Scenarios: Stack Exchange, Quora (Kaggle), education

- Profile: Computer Scientist
- Team: Iñigo, Eneko, Montse Oier, Aitor

Thanks!

e.agirre@ehu.eus

@eagirre

Disambiguation of words and entities

Three of the greatest guitarists started their career in a single band: Clapton, Beck and Page.

- Linking text to entities in knowledge bases
- Deep learning and graph-based approaches
- Integrate many Kbs: Wikipedia, WordNet, etc.
- Multilingual

Profile: Computer scientist
Team: Aitor, Eneko, Ander, Oier, Josu, Mikel, Gorka.