

# Basics and (Modern) Methods for Natural Language Processing

*Digital Product School of UnternehmerTUM*  
*June 5, 2018*

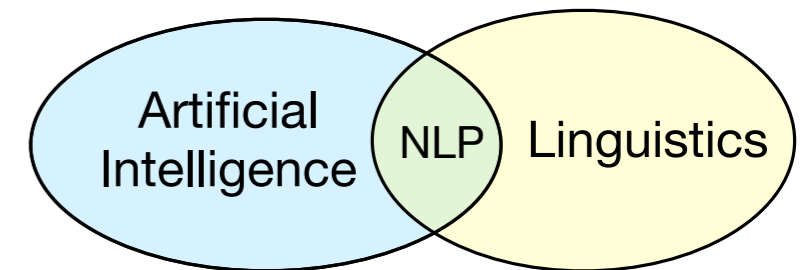
Nikolaos Pappas  
Natural Language Understanding Group  
Idiap Research Institute, Martigny

# Outline of the talk

1. Introduction and Motivation
2. Word Representation Learning
  - Semantic similarity
  - Traditional and recent approaches
  - Intrinsic and extrinsic evaluation
3. Word Sequence Modeling
  - Essentials: Encoders, Attention, DL tricks
  - Text Classification
  - Machine Translation
4. Conclusion and Discussion

# Natural Language Processing

- NLP is a field at the intersection of AI and linguistics
  - Linguistics (structure of language, brain mappings, language learning)
  - Computational Linguistics (comp. models of language, tools for studying language)
- **Goals**
  - ✓ Process large amounts of natural text
  - ✓ Give computers the ability to “understand” language to perform useful tasks
    - ➔ Intrinsic tasks: parsing, language modeling, etc
    - ➔ Extrinsic tasks: speech recognition, translation, etc



# Levels of processing

## ➔ Lexical level

- Speech: phonetic analysis
- Vision: character recognition

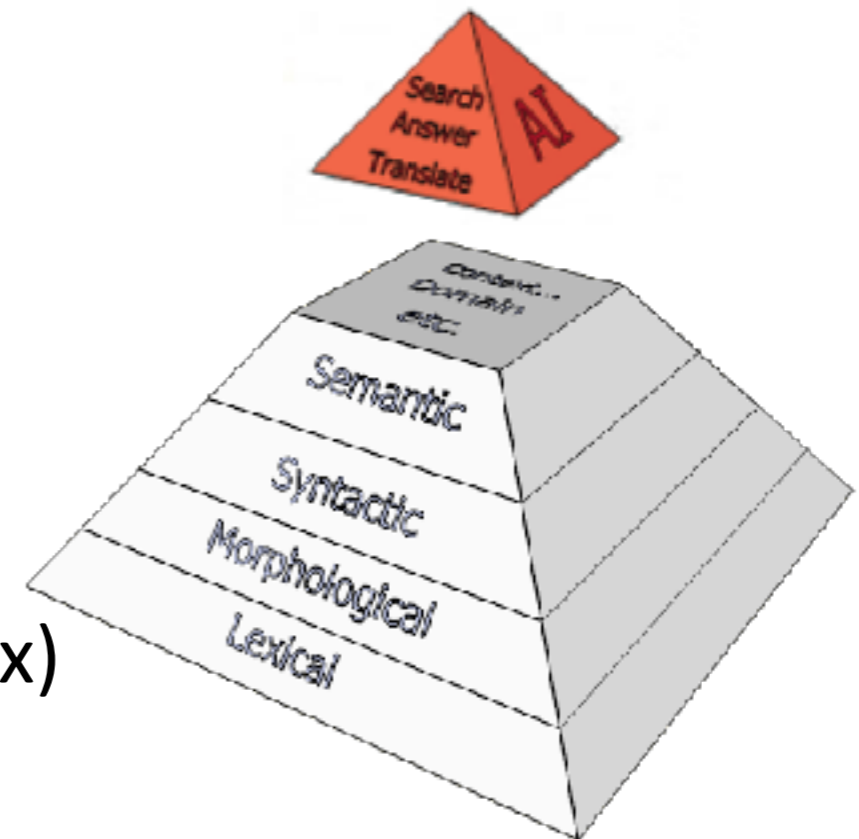
## ➔ Morphological & Syntactic levels

- Word structure (forms, inflections)
- Sentence structure (grammar, syntax)

## ➔ Semantic & Discourse levels

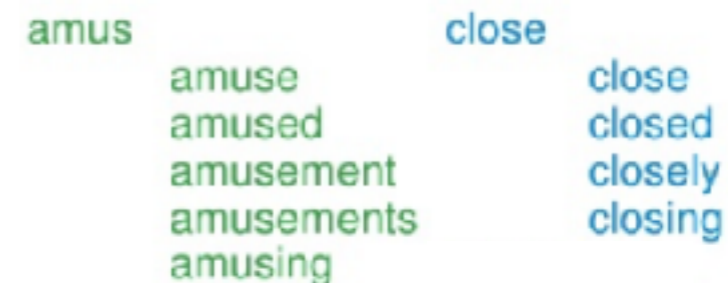
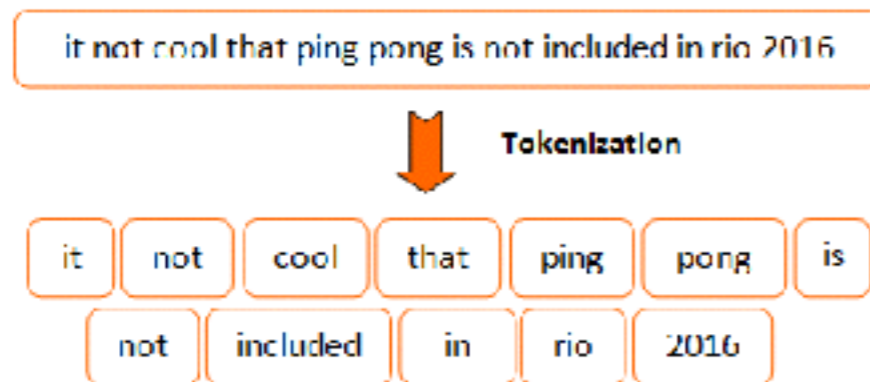
- Word and sentence meanings
- Broad context, co-reference

➔ The ultimate goal of a system however is to be able translate, assist, retrieve, classify, communicate



# Intrinsic tasks: Text segmentation & Morphology

- Tokenization (split text into meaningful segments)
- Stemming (reduction of word forms to stems)



- Punctuation prediction

>> Original speech utterance:

you are quite welcome and by the way we may get other reservations so could you please call us as soon as you fix the date

>> Punctuated (and cased) version:

You are quite welcome . And by the way , we may get other reservations , so could you please call us as soon as you fix the date ?

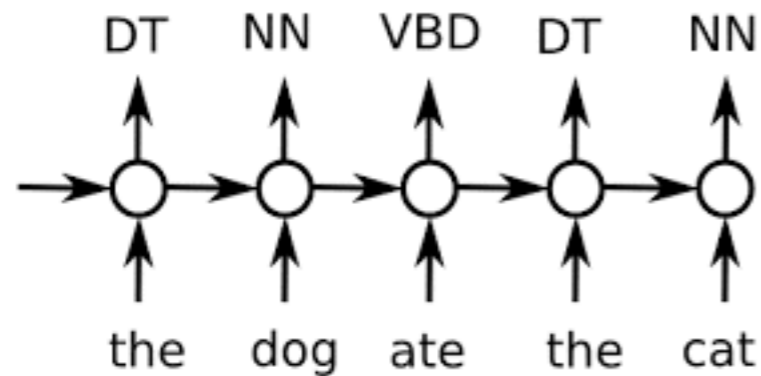
- Lemmatization (reduction of word forms to base form & intended POS and meaning )

l:happy	happy, happier, happiest
l:go	go, goes, going, gon(na), went, gone
l:man	man, men

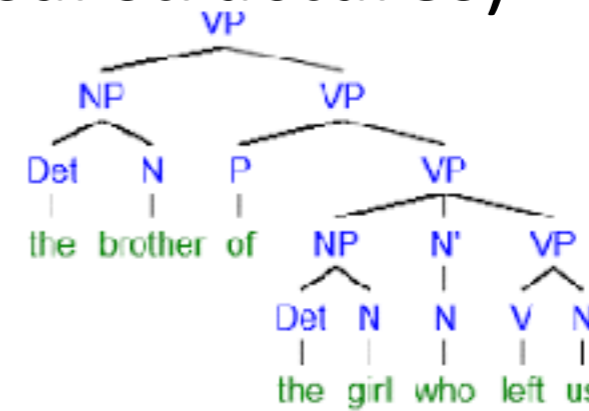
**Morphemes:** smallest linguistic pieces with a grammatical function (inflectional: fish-> fishes, derivational: fish -> fishery, compounding: sky + scraper -> skyscraper)

# Intrinsic tasks: Syntax & Grammar

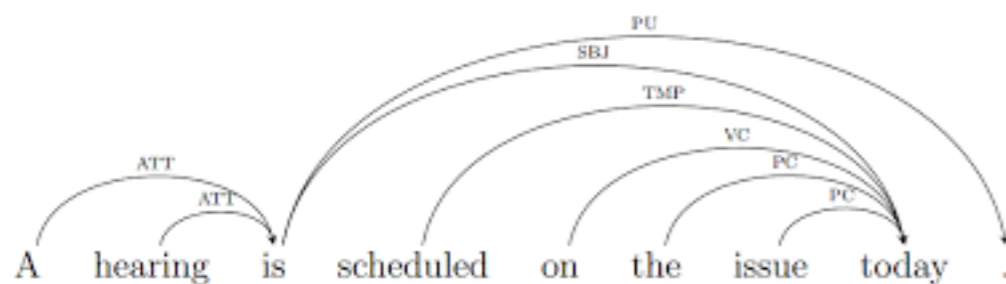
- Part-of-speech tagging (POS tags sequence)



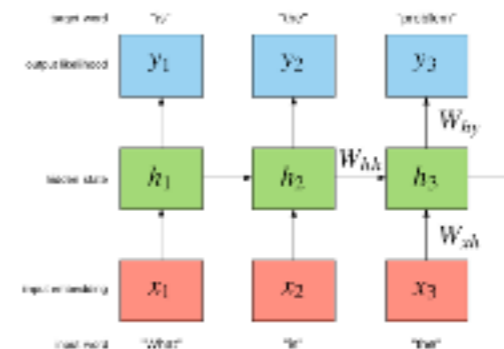
- Constituency parsing (nested phrasal structures)



- Dependency parsing (role specific structures)



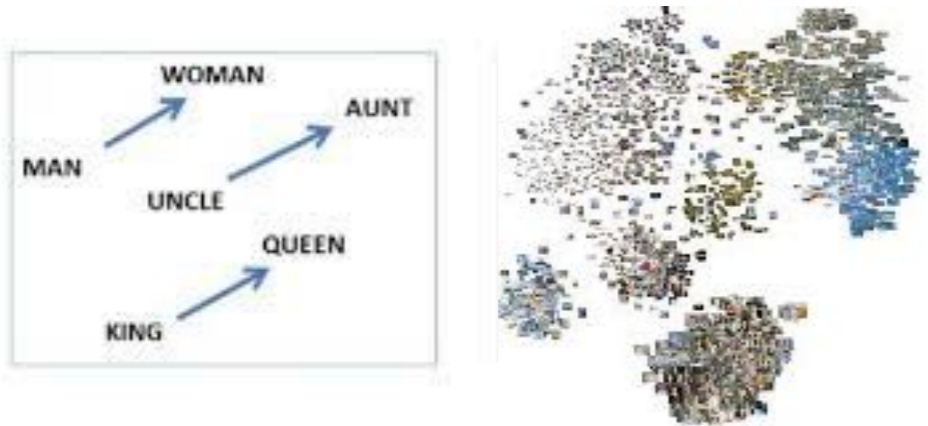
- Language modeling (word sequence)



**DT:** determiner, **NN:** noun, singular, **VBD:** verb past tense, **NP:** noun phrase, **VP:** Verb phrase, **ATT:** attributive, **SBJ:** nominal subject, **TMP:** temporal modifier, **PC:** prepositional complement

# Intrinsic tasks: Semantics & Discourse

- Lexical semantics



- Textual entailment (directional relation between text fragments)

$P^a$	A senior is waiting at the window of a restaurant that serves sandwiches.	Relationship
$H^b$	A person waits to be served his food.	Entailment
	A man is looking to order a grilled cheese sandwich.	Neutral
	A man is waiting in line for the bus.	Contradiction
<sup>a</sup> P, Premise.		
<sup>b</sup> H, Hypothesis.		

- Named entity recognition

Sheryl Sandberg hits back at Apple CEO Tim Cook over his past jobs at Facebook

Potential tags:

ORGANIZATION

LOCATION

PERSON

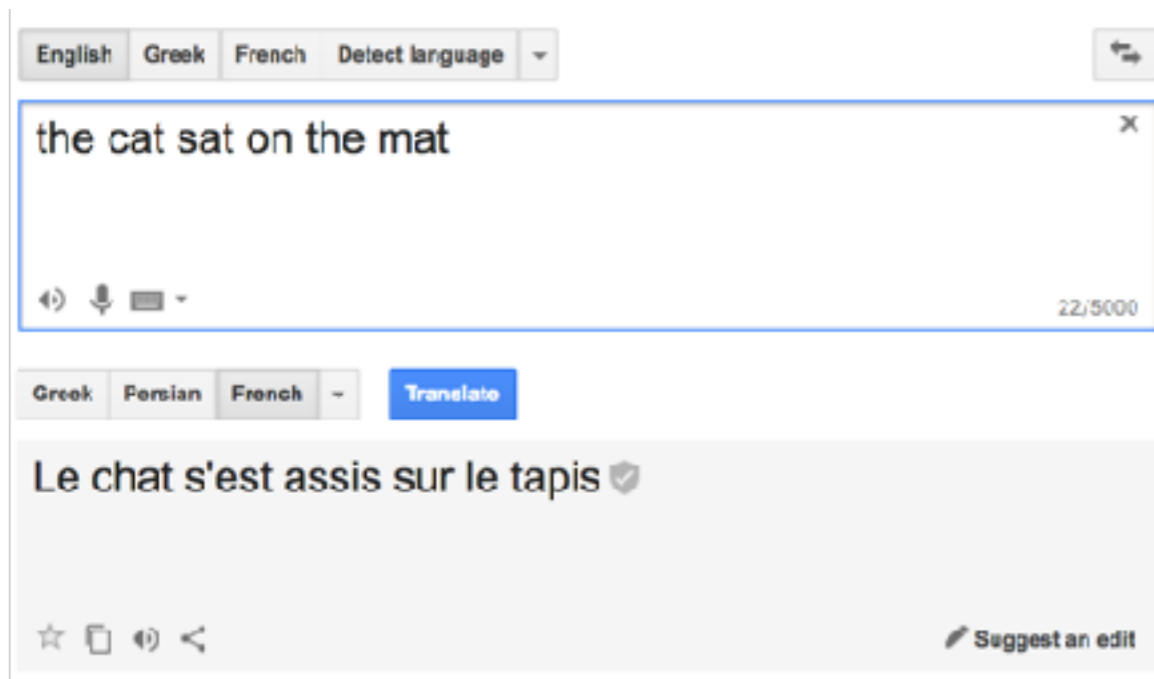
- Coreference Resolution (find expressions referring to the same entity in a text)

*"I voted for Nader because he was most aligned with my values," she said.*



# Extrinsic Tasks

- Machine translation



- Question answering

## Question

What does the physics engine allow for?

## Passage Context

Kerbal Space Program (KSP) is a space flight simulation video game developed and published by Squad for Microsoft Windows, OS X, Linux, PlayStation 4, Xbox One, with a Wii U version that was supposed to be released at a later date. The developers have stated that the gaming landscape has changed since that announcement and more details will be released soon. In the game, players direct a nascent space program, staffed and crewed by humanoid aliens known as "Kerbals". The game features a realistic orbital physics engine, allowing for **various real-life orbital maneuvers** such as Hohmann transfer orbits and bi-elliptic transfer orbits.



# Extrinsic Tasks

- Sentiment analysis

Prod:

The hotel is really beautiful. Moviestar feeling and decadence from yesterday. The pool is designed by Johnny Weissmuller. So it was a trendy pool. The food at the restaurant was really good. Very nice and helpful service at the frontesk.

Cons: this is what made my grade a 3 instead of 4. We had problems to get the wi-fi working. If you're not depend this is not interesting. We talked several times with the front desk.

When we're there they had party event in the pool area between noon and 5 PM. The pool area was occupied with young party animals. So the area wasn't fun for UD.



- Summarization

cia documents reveal iot-specific televisions can be used to secretly record conversations . cybercriminals who initiated the attack managed to commandeer a large number of internet-connected devices in current use . cia documents revealed that microwave ovens can spy on you - maybe if you personally don't suffer the consequences of the sub-par security of the iot .

Internet of Things (IoT) security breaches have been dominating the headlines lately . WikiLeaks's leak of CIA documents revealed that internet-connected televisions can be used to secretly record conversations . Trump's advisor Kellyanne Conway believes that microwave ovens can spy on you - maybe she was referring to microwave cameras which indeed can be used for surveillance . And don't delude yourself that you are immune to IoT attacks , with 98 % of security professionals responding to a new survey expecting an increase in IoT breaches this year . Even if you personally don't suffer the consequences of the sub-par security of the IoT , your connected devices may well be unwittingly cooperating with criminals . Last October , Internet service provider Dyn came under an attack that disrupted access to popular websites . The cybercriminals involved in the attack managed to commandeer a large number of IoT devices (referred to as botnets) mostly DVRs and cameras to serve as their helpers . As a result , cybersecurity expert Bruce Schneier has called for government regulation of the IoT , concluding that both IoT manufacturers and their customers don't care about the security of the 8.4 billion internet-connected devices in current use . Whether because of government legislation or good old-fashioned self-interest , we can expect increased investment in IoT security technologies . In its recently-released TechRadar report for security and risk professionals , Forrester Research discusses the outlook for the 13 most relevant and important IoT security technologies , warning that "there is no single , magic security bullet that can easily fix all IoT security issues ." Listed on Forrester's analysis , here's my list of the 6 hottest technologies for IoT security : IoT network security : Protecting and securing the network connecting IoT devices to back-end systems on the internet . IoT network security is a bit more challenging than traditional network security because there is a wider range of communication protocols , standards , and device capabilities , all of which pose significant issues and increased complexity . Key capabilities include traditional endpoint security features such as antivirus and antimalware as well as other features such as firewalls and intrusion prevention and detection systems . Sample vendors : Ezyntec Networks , Cisco , Gateworks , and Sonos . IoT authentication : Providing the ability for users to authenticate an IoT device . Including manual methods (such as a physical key) such as a connected car 1 , means from simple one-way radio to more sophisticated authentication mechanisms such as two-factor

- Dialogue agents / Chatbots
- Topic recognition
- Search and retrieval
- and more

# What is special about natural language?

- Formal languages are **static**, **explicit** and **non-ambiguous**
  - Defined as mathematical abstractions (alphabet, rules)
  - One can explicitly enumerate all well-formed words
- Natural languages are **dynamic**, **implicit** and **ambiguous**
  - Exist in the real world and is spoken by its users
  - Grammar is discovered through empirical investigation

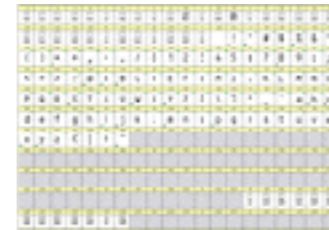
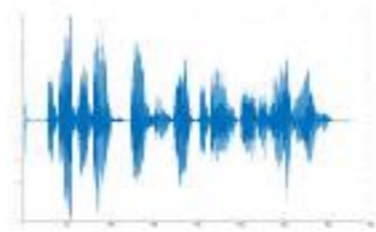
“Cathrine and John gave flowers to Mary.  
She said “thanks” and put them in a vase.”

“I need to talk to you asap.” (abbreviations)

“Did you download the app?” (neologisms)

# What is natural language?

- A naturally evolved system used by humans to express thoughts for
  - (i) communicating with one another, (ii) learning from previous
  - (iii) experiences and achieving their goals
- Essentially, it is a discrete / symbolic / categorical signaling system
  - ✓ Symbols are **invariant** across signals (audio, visual)



- ✓ Concise and grounded on shared knowledge (entails ambiguities)

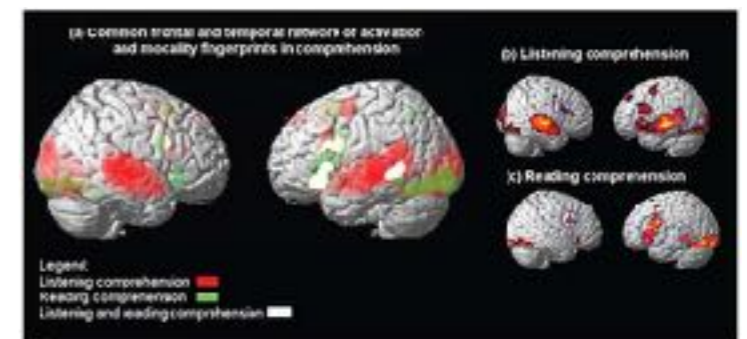
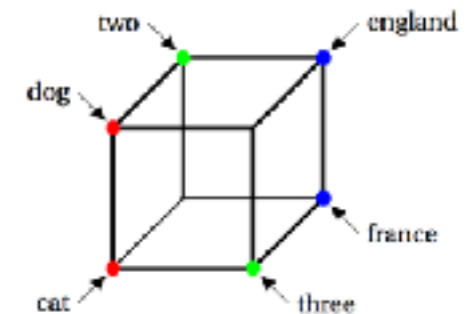
“Did you watch the finals? Our goalkeeper was useless!”

- ✓ Unlimited expressive power (implies flexible interpretation rules  
i.e. meaning cannot be exclusively expressed in the surface form)

“All politicians lie.” |  $\forall x, \text{politician}(x) \Rightarrow \text{liar}(x)$

# What are the main challenges in language `understanding`?

- Symbolic encodings require large vocabularies
  - Sparsity issues for machine learning
  - Scaling issues in real-world settings
- Brain encodings appear to be a continuous pattern of activations (distributed across neurons)
  - ➔ Continuous encodings provides a cognitively plausible way to encode thoughts



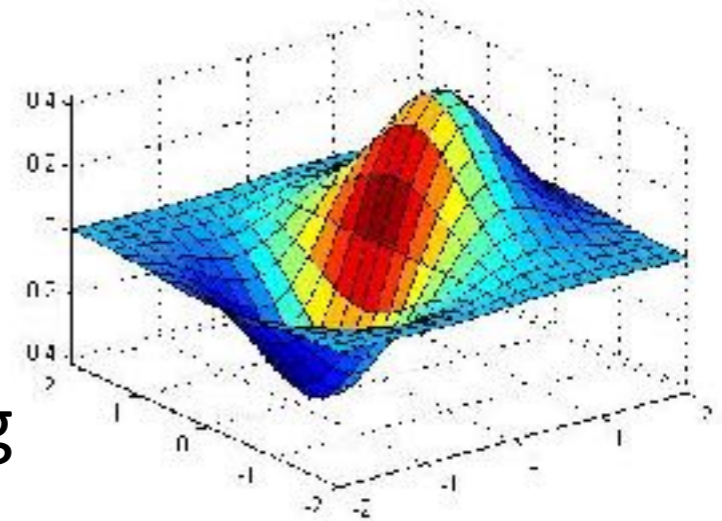
Buchweitz et al. 2009

- **Challenges**
  - How to learn continuous encodings that generalize well?
  - Can we encode very complex thoughts in a single continuous encoding?
  - Can we create and reason over thoughts to solve any NLP task?
  - How to transfer knowledge from one domain, task, language to another?

# What is Deep Learning?

- Machine Learning boils down to minimizing an objective function to increase task performance

- Mostly relies on human-crafted features 😞
- Tasks involve regression, classification, structured prediction, representation learning



→ **Representation Learning:** Learn good features or representations

→ **Deep Learning:** Machine learning algorithms based on multiple levels of representation or abstraction

- ✓ Biologically inspired from how the human brain works
- ✓ Neurons activate to certain inputs and excite other neurons
- ✓ Can handle a variety of input, such as vision, speech, and language

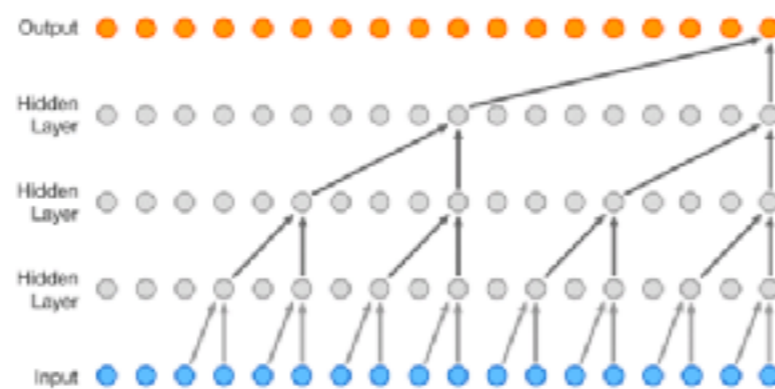
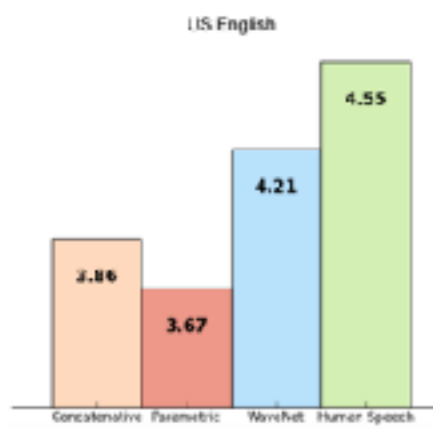


# Deep Learning: Why this decade?

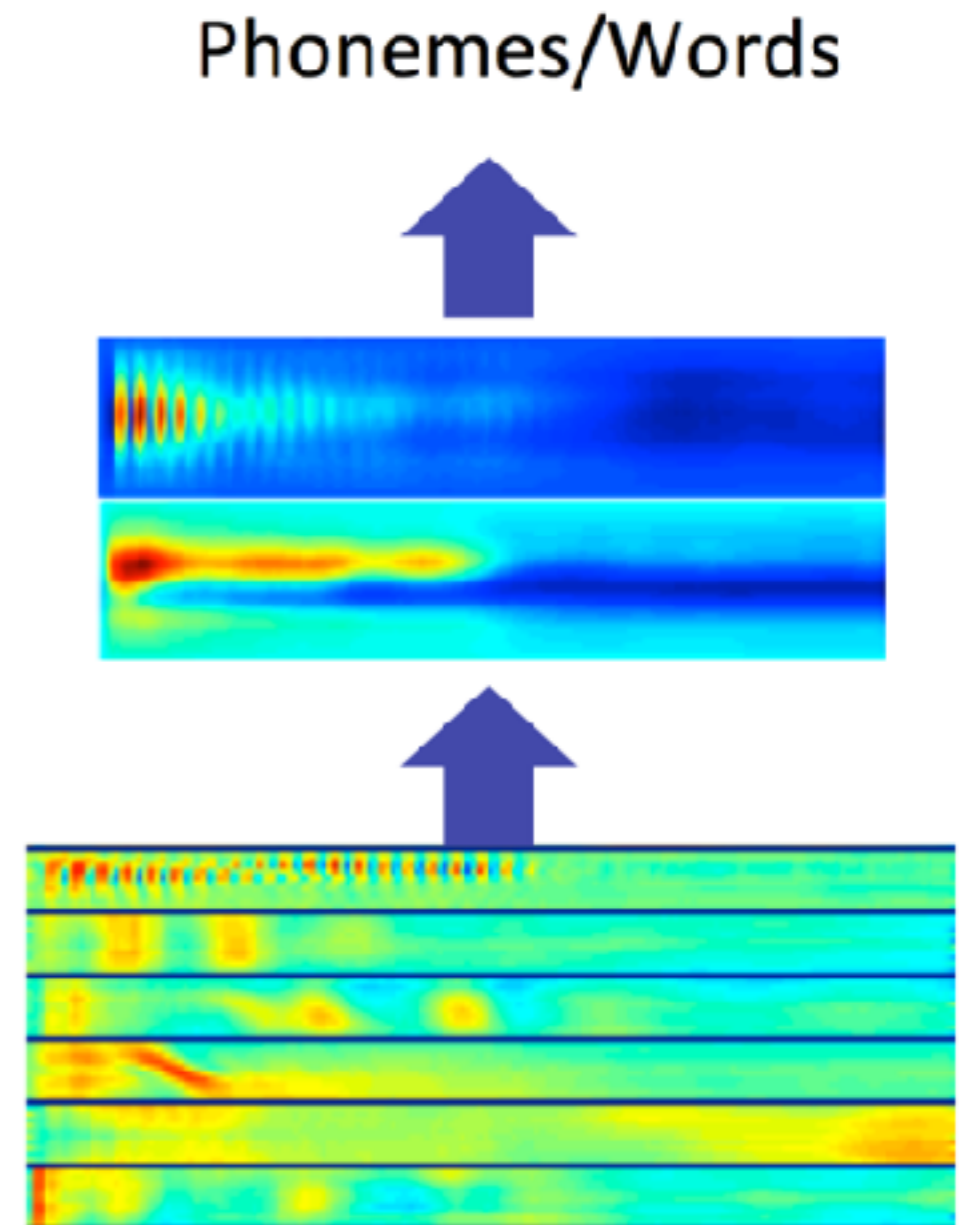
- What enabled deep learning techniques to start outperforming other machine learning techniques since [Hinton et al. 2006](#)?
- Larger amounts of data
- Faster computers and multicore cpu and gpu
- New models, algorithms and improvements over “older” methods ([speech](#), [vision](#) and [language](#))

# Deep Learning for speech: Phoneme detection

- The first breakthrough results of “deep learning” on large datasets by [Dahl et al. 2010](#)
  - -30% reduction of error
- Most recently on speech synthesis [Oord et al. 2016](#)



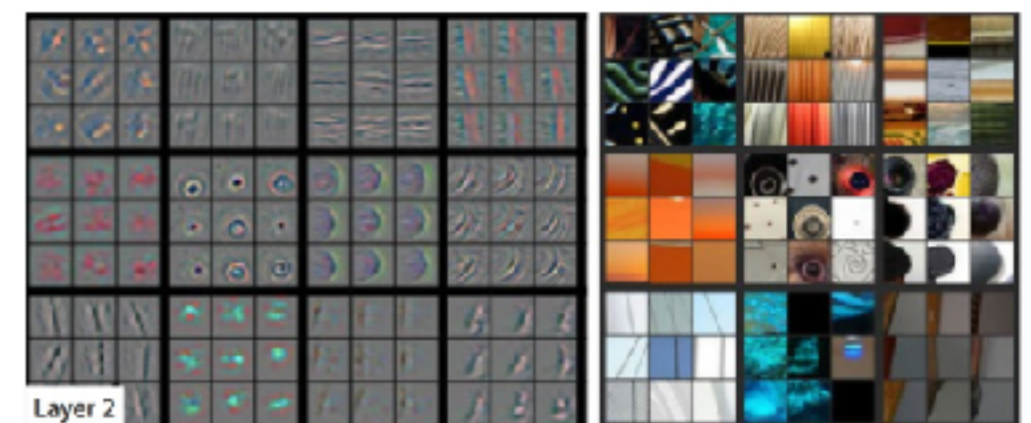
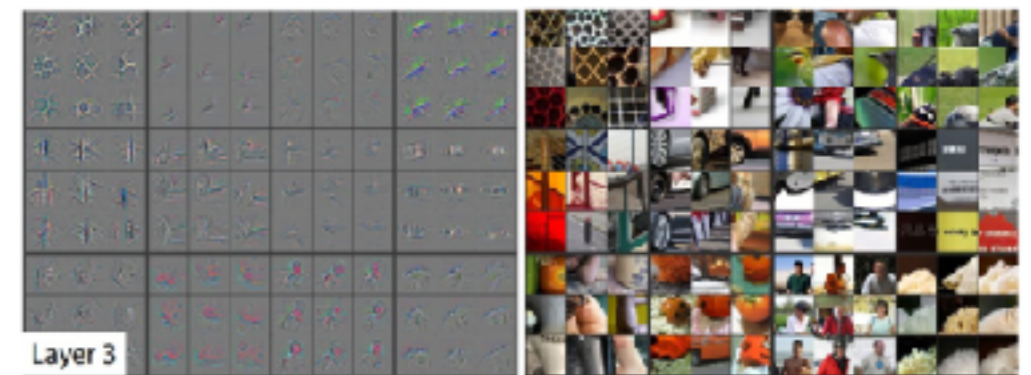
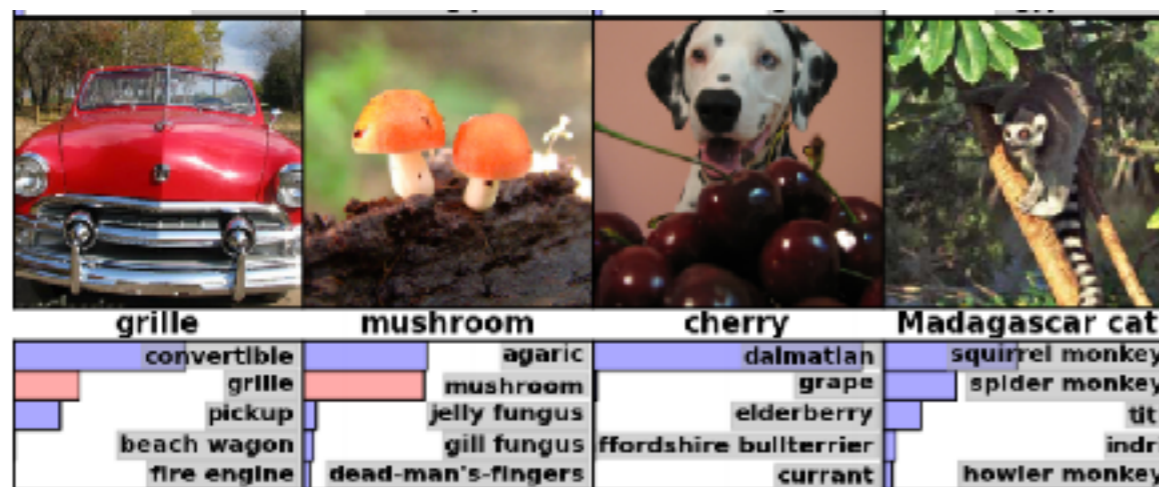
Nikolaos Pappas






# Deep Learning for vision: Object detection

- Popular topic for DL
- Breakthrough on ImageNet by [Krizhevsky et al. 2012](#)
  - -21% and -51% error reduction at top 1 and 5



Zeiler and Fergus (2013)

# Deep Learning for language: Ongoing

- Significant improvements in recent years across different levels (phonology, morphology, syntax, semantics) and applications in NLP
  - **Machine translation** (most notable) 
  - **Question answering**
  - **Sentiment classification**
  - **Summarization**

Still a lot of work to be done...  
(beyond supervised and “basic” recognition)

# Deep Learning for language: Machine Translation

- Reached the state-of-the-art in one year: [Bahdanau et al. 2014](#), [Jean et al. 2014](#), [Gulcehre et al. 2015](#)

(a) English→French (WMT-14)

	<b>NMT(A)</b>	Google	P-SMT
NMT	32.68	30.6*	<b>37.03*</b>
+Cand	33.28	—	
+UNK	33.99	32.7°	
+Ens	<b>36.71</b>	<b>36.9°</b>	

(b) English→German (WMT-15)

Model	Note
<b>24.8</b>	Neural MT
24.0	U.Edinburgh, Syntactic SMT
23.6	LIMSI/KIT
22.8	U.Edinburgh, Phrase SMT
22.7	KIT, Phrase SMT

(c) English→Czech (WMT-15)

Model	Note
<b>18.3</b>	Neural MT
18.2	JHU, SMT+LM+OSM+Sparse
17.6	CU, Phrase SMT
17.4	U.Edinburgh, Phrase SMT
16.1	U.Edinburgh, Syntactic SMT

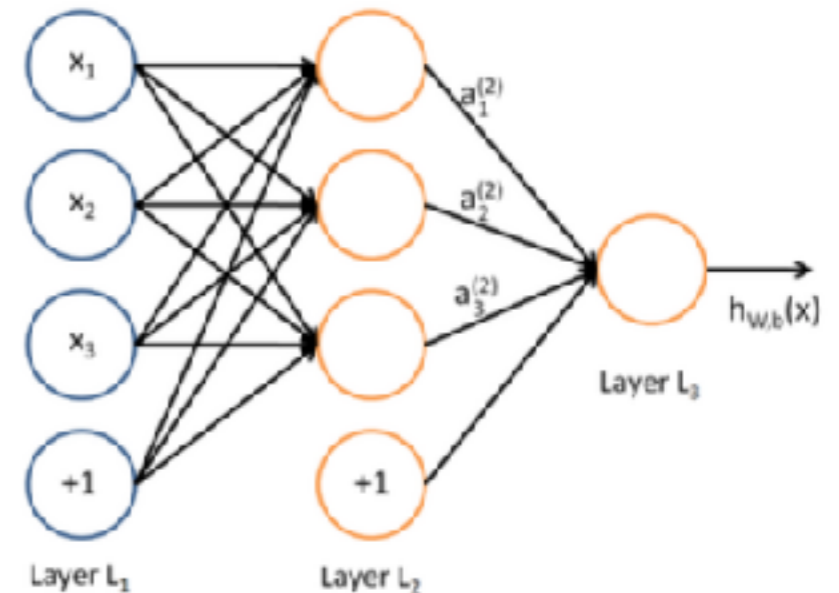
# Neural network components for language `understanding`

- **Distributed Representations (word/subword units)**
  - Ability to represent `meaning` efficiently
- **Abstraction & Composition (word sequences)**
  - Ability to compose complex `meanings` from simpler ones
- **Attention Mechanism**
  - Ability to focus on/collect what is `relevant` (input, memory)
- **Memory Mechanism**
  - Ability to store/retrieve important previous information/knowledge
- **Reasoning Mechanism**
  - Ability to reason with what is `relevant`
- **Learning Mechanism**
  - Ability to learn from past experience

...

# Outline of the talk

1. Introduction and Motivation
  - **Basics: Perceptron, NNs, SGD**
2. Word Representation Learning
  - Semantic similarity
  - Traditional and recent approaches
  - Intrinsic and extrinsic evaluation
3. Word Sequence Modeling
  - Essentials: Encoders, Attention, DL tricks
  - Text Classification
  - Machine Translation
4. Conclusion and Discussion



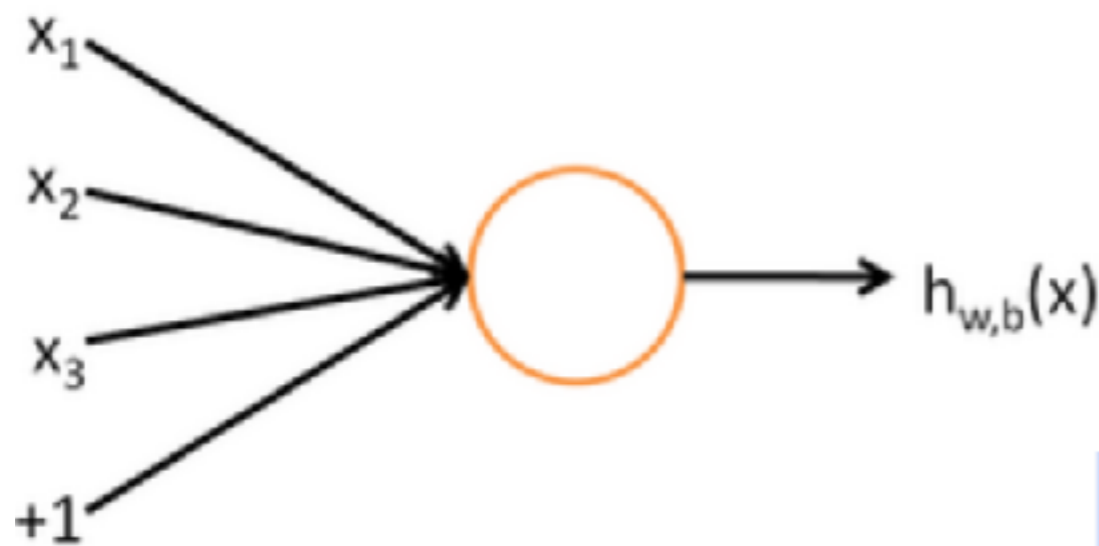
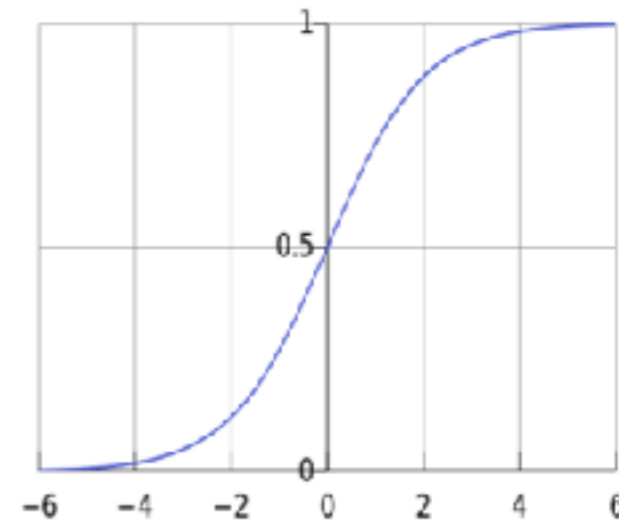


# Basics: Perceptron

$$h_{w,b}(x) = f(w^T x + b)$$

$b$ : We can have an “always on” feature, which gives a class prior, or separate it out, as a bias term

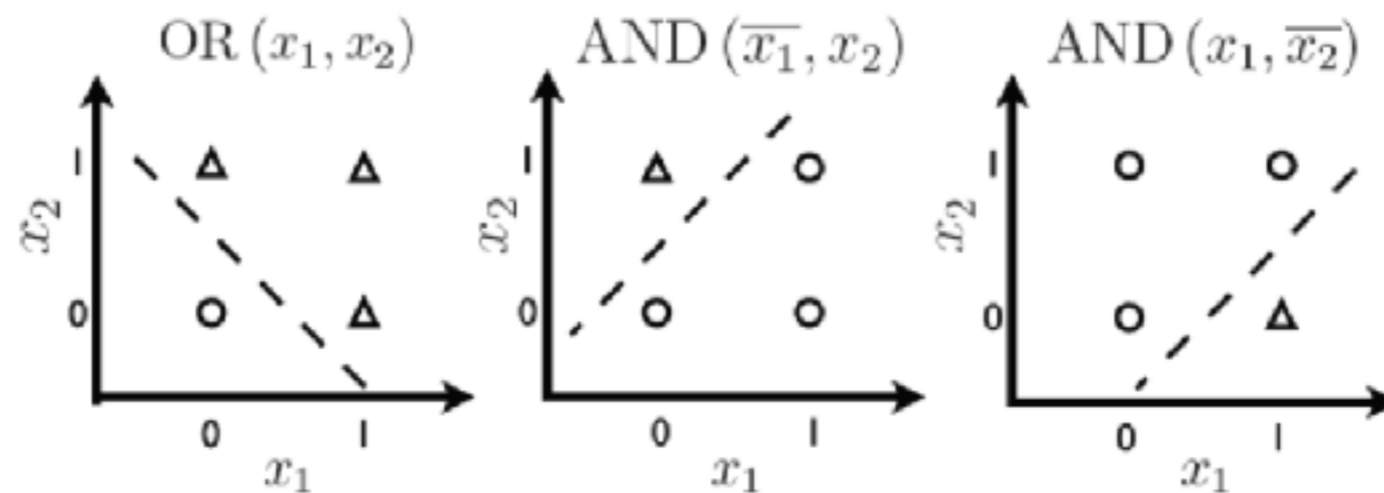
$$f(z) = \frac{1}{1 + e^{-z}}$$



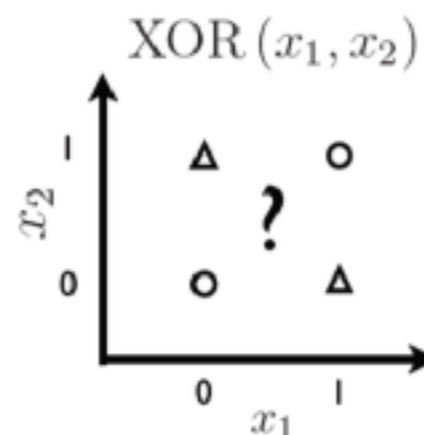
$w, b$  are the parameters of this neuron  
i.e., this logistic regression model

# Basics: What can a perceptron do?

- Solve linearly separable problems



- ... but not non-linearly separable ones.



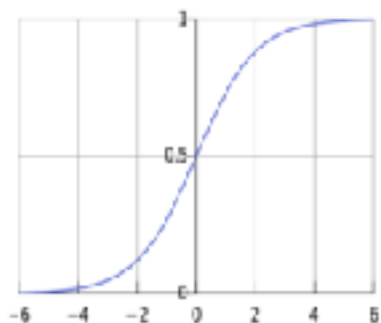


# Basics: From logistic regression to neural networks

Vector form: 
$$P(c | d, \lambda) = \frac{e^{\lambda^T f(c,d)}}{\sum_{c'} e^{\lambda^T f(c',d)}}$$

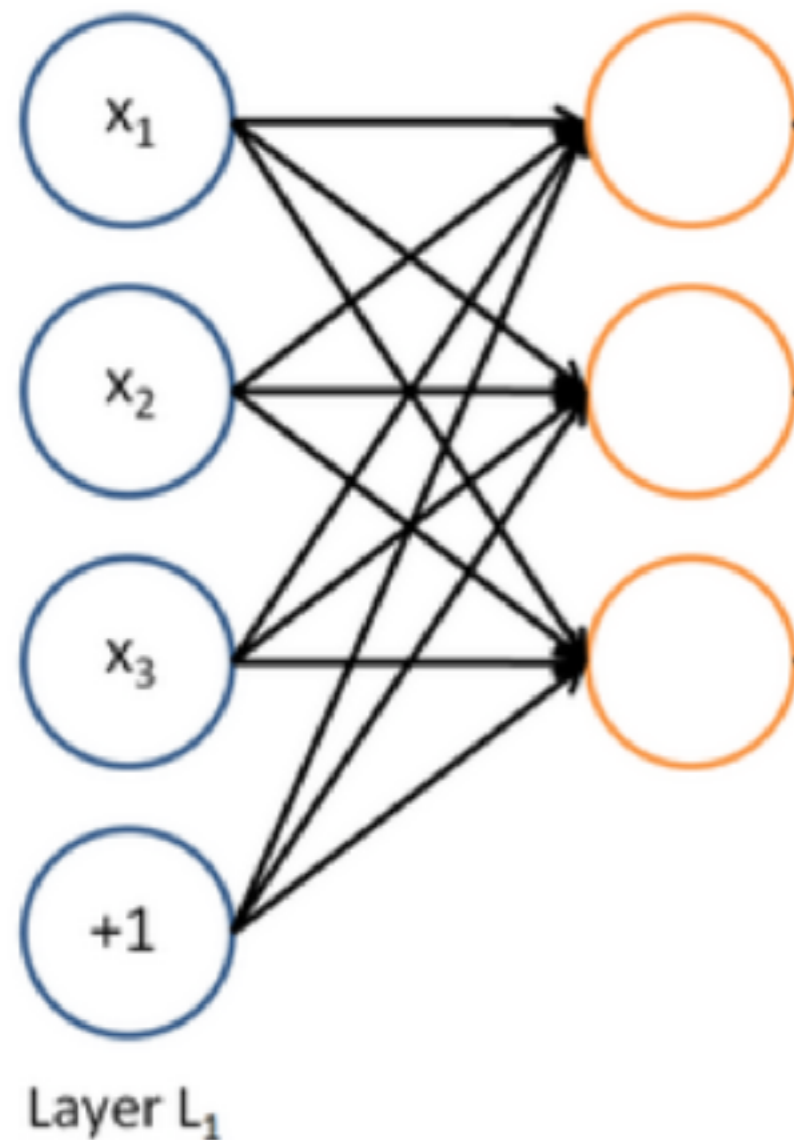
Make two class:

$$\begin{aligned}
 P(c_1 | d, \lambda) &= \frac{e^{\lambda^T f(c_1,d)}}{e^{\lambda^T f(c_1,d)} + e^{\lambda^T f(c_2,d)}} = \frac{e^{\lambda^T f(c_1,d)}}{e^{\lambda^T f(c_1,d)} + e^{\lambda^T f(c_2,d)}} \cdot \frac{e^{-\lambda^T f(c_1,d)}}{e^{-\lambda^T f(c_1,d)}} \\
 &= \frac{1}{1 + e^{\lambda^T [f(c_2,d) - f(c_1,d)]}} = \frac{1}{1 + e^{-\lambda^T x}} \quad \text{for } x = f(c_1,d) - f(c_2,d) \\
 &= f(\lambda^T x)
 \end{aligned}$$



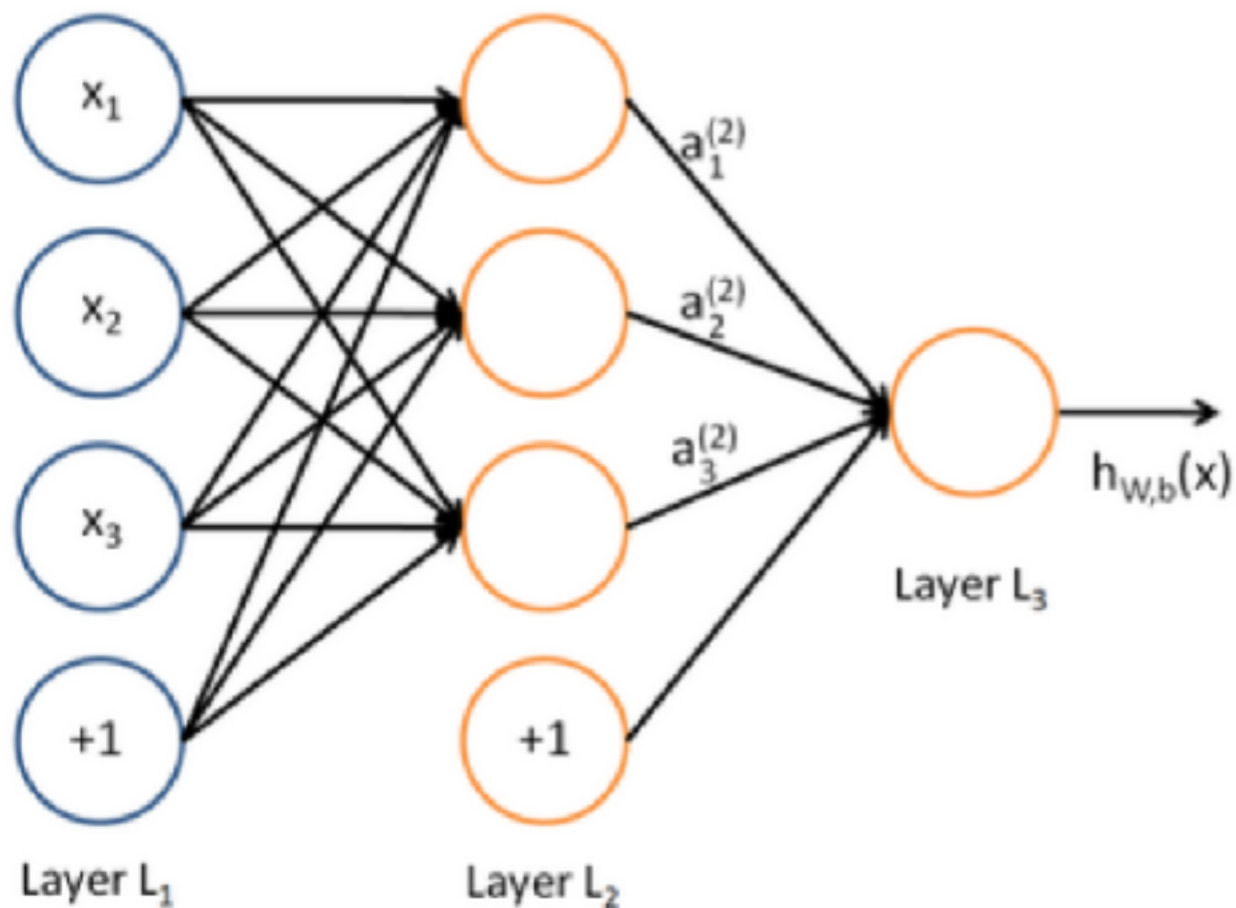
for  $f(z) = 1/(1 + \exp(-z))$ , the logistic function – a sigmoid **non-linearity**.

# Basics: Neural network



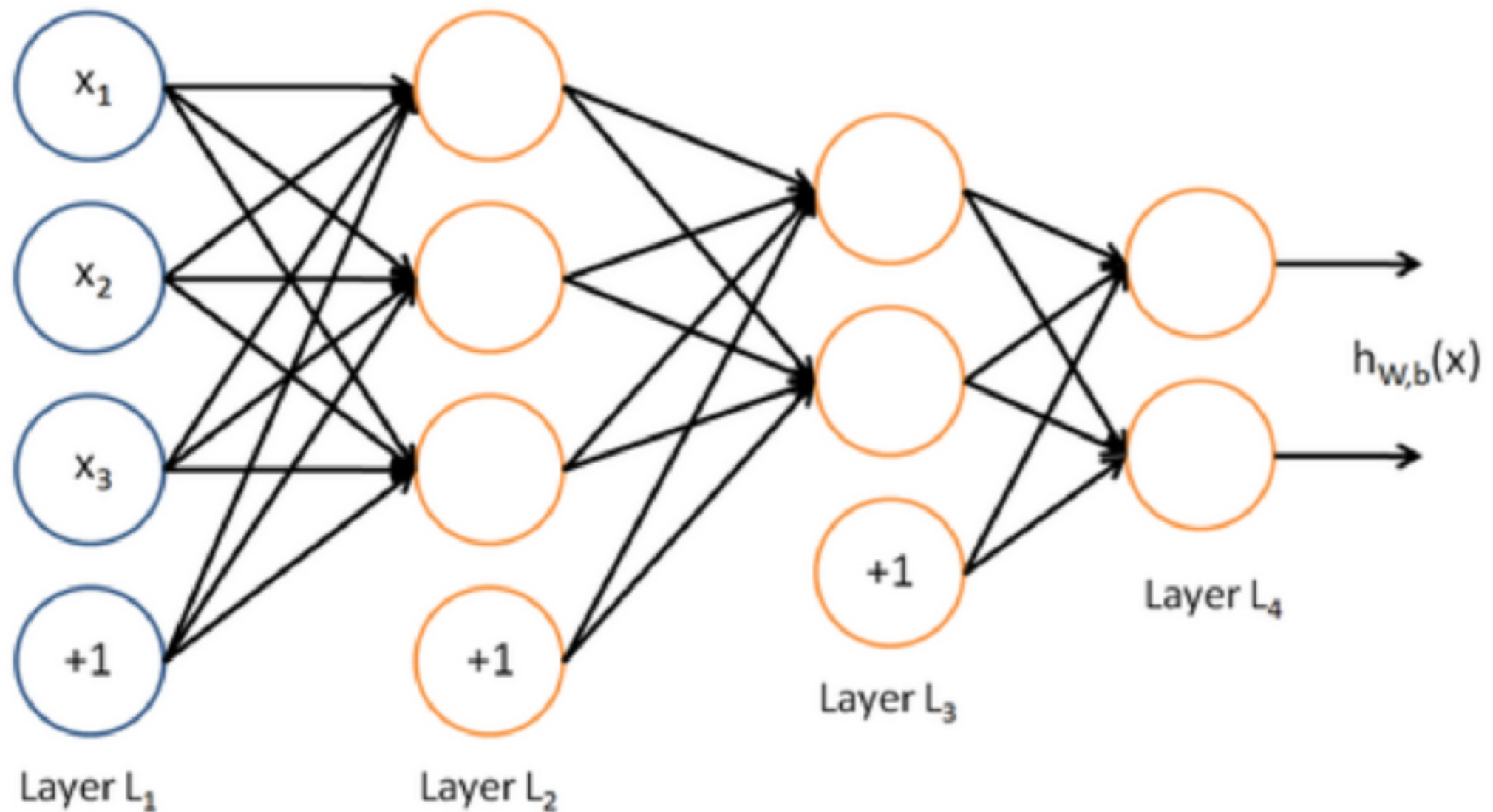
- Apply several regressions to obtain a vector of outputs
- The values of the outputs are initially unknown
  - No need to specify ahead of time what values the logistic regressions are trying to predict

# Basics: Neural network



- The intermediate variables are learned directly based on the training objective
- This makes them do a good job at predicting the target for the next layer
- **Result:** able to model non-linearities in the data!

# Basics: Neural network with multiple layers



# Basics: Learning model parameters with gradient descent

- Given training data  $\mathcal{D} = \{x^{(i)}, y^{(i)}\}_{i=1}^N$  find  $W$  and  $b$  that minimizes loss with respect to these parameters

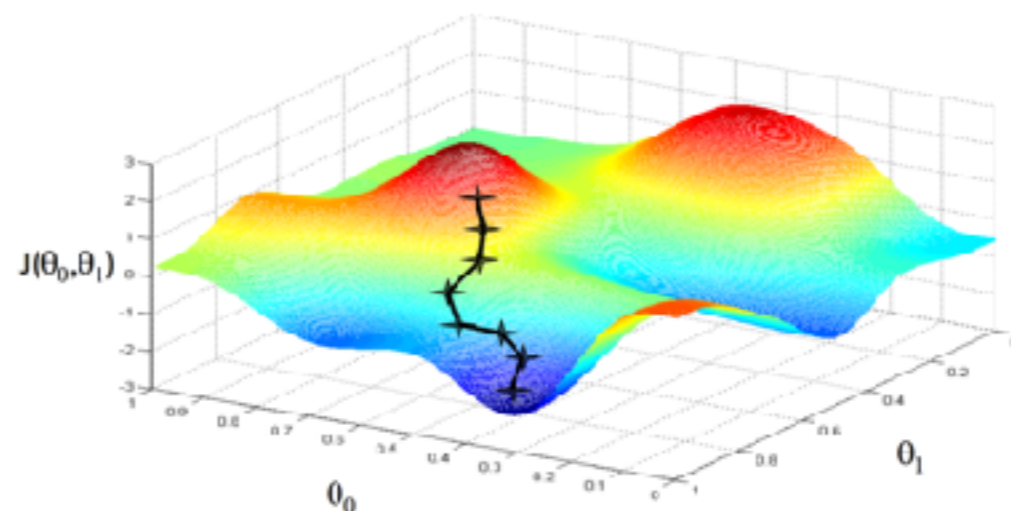
$$\mathcal{J}(\theta) = \frac{1}{2N} \sum_{i=1}^N (g(a(x^{(i)})) - y^{(i)})^2$$

- Compute gradient with respect to parameters and make small step towards the direction of the negative gradient
  - Apply chain-rule for nested functions e.g.  $y=f(g(x))$

$$W = W - \alpha \frac{\partial \mathcal{J}}{\partial W}$$

$$b = b - \alpha \frac{\partial \mathcal{J}}{\partial b}$$

$\alpha$  - learning rate or step size.



# Basics: Stochastic gradient descent (SGD)

- Approximate the gradient using a mini-batch of examples instead of entire training set
- Online SGD when mini batch size is one
- Most commonly used when compared to GD

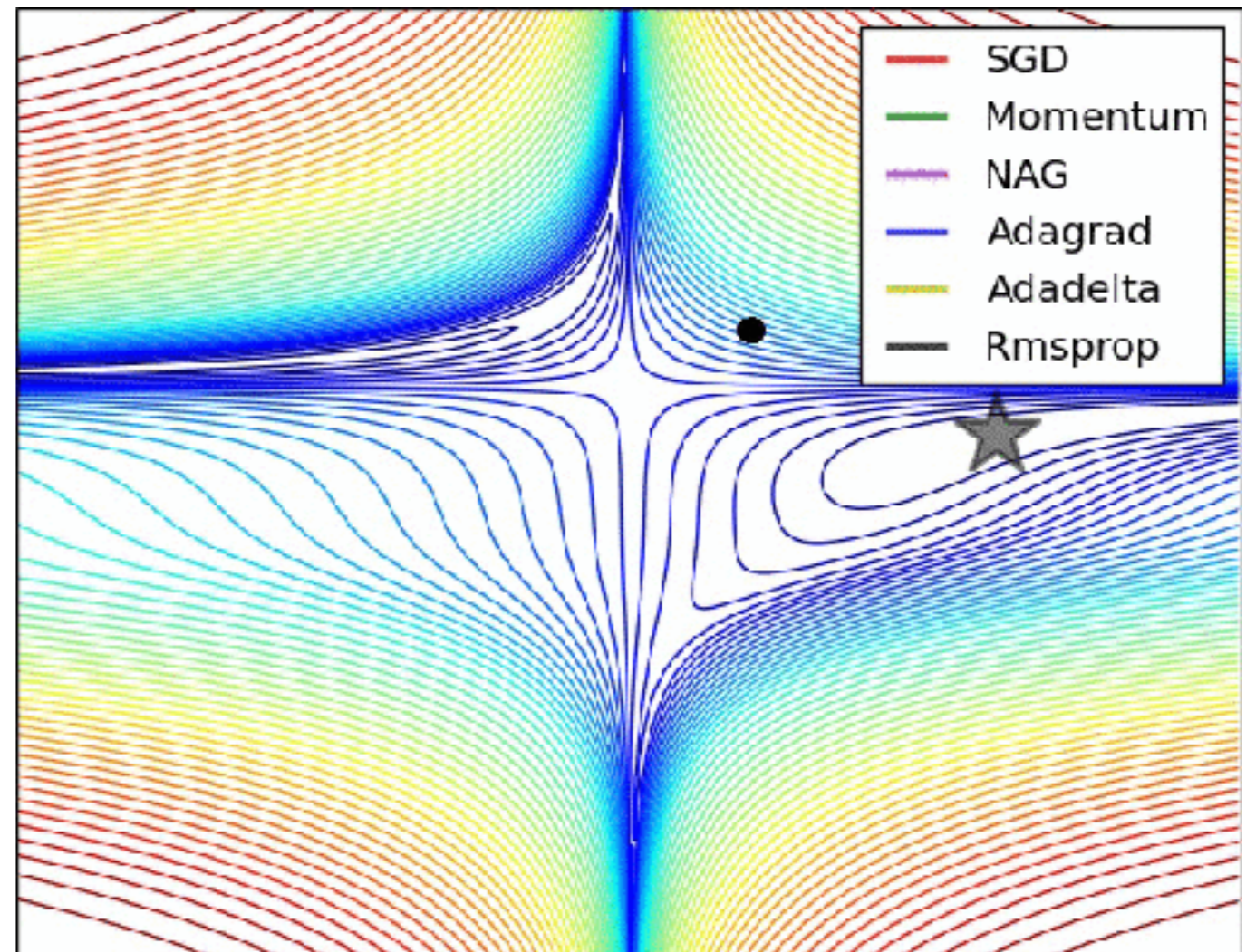
$$w_k = w_k - \alpha \cdot (g(a(x^{(i)})) - y^{(i)}) \cdot g'(a(x^{(i)})) \cdot x_k^{(i)} \quad \text{for } k = 1, \dots, d$$

$$b = b - \alpha \cdot (g(a(x^{(i)})) - y^{(i)}) \cdot g'(a(x^{(i)}))$$



# Basics: Choosing a Stochastic Optimization Algorithm

- Several out-of-the-box strategies for decaying learning rate of an objective function:
  - Select the best according to validation set performance





# Training neural networks with arbitrary layers: Backpropagation

- We still minimize the objective function but this time we “backpropagate” the errors to all the hidden layers
- Chain rule: If  $y = f(u)$  and  $u = g(x)$ , i.e.  $y=f(g(x))$ , then:

$$\frac{dy}{dx} = \frac{dy}{du} \frac{du}{dx} = \frac{df(u)}{du} \frac{dg(x)}{dx}$$

- Useful basic derivatives:

$$\frac{\partial \mathbf{x}^T \mathbf{a}}{\partial \mathbf{x}} = \frac{\partial \mathbf{a}^T \mathbf{x}}{\partial \mathbf{x}} = \mathbf{a}$$

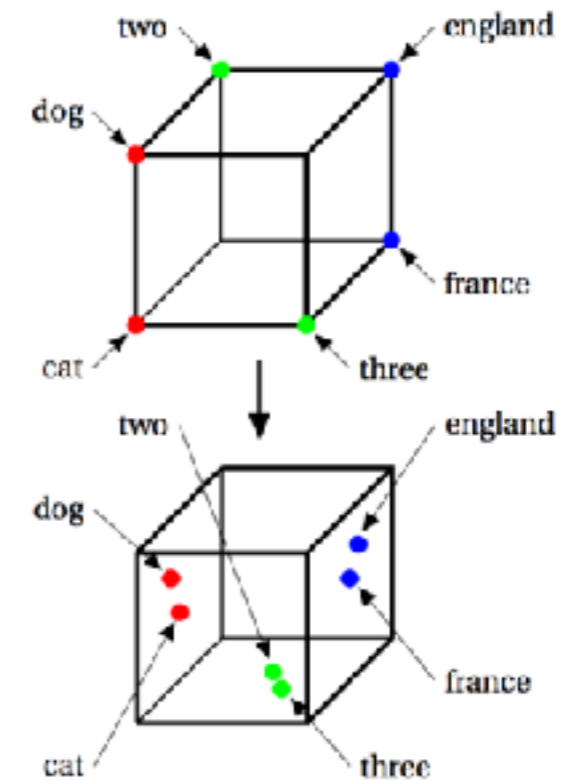
Typically, backprop computation is implemented in popular libraries: **Theano, Torch, Tensorflow**

# Basics: The end

- Essentially, now we have all the basic “ingredients” we need to build deep neural networks
- However, we will also need
  - ➔ Ability to learn from different inputs (spatial, sequential, continuous vs discrete)
  - ➔ Overcome optimization difficulties (exploding/vanishing gradient, information flow, convergence)
  - ➔ Avoid overfitting / Regularization (dropout, L2 norm)
  - ➔ and other...

# Outline of the talk

1. Introduction and Motivation
2. Word Representation Learning
  - Semantic similarity
  - Traditional and recent approaches
  - Intrinsic and extrinsic evaluation
3. Word Sequence Modeling
  - Essentials: Encoders, Attention, DL tricks
  - Text Classification
  - Machine Translation
4. Conclusion and Discussion



\* Figure from Lebret's thesis, EPFL

# Semantic similarity: How similar are two linguistic items?

- **Word level**

*screwdriver* —?—> *wrench*  
*screwdriver* —?—> *hammer*  
*screwdriver* —?—> *technician*  
*screwdriver* —?—> *fruit*

very similar  
little similar  
related  
unrelated

- **Sentence level**

*The boss fired the worker*  
*The supervisor let the employee go*  
*The boss reprimanded the worker*  
*The boss promoted the worker*  
*The boss went for jogging today*

very similar  
little similar  
related  
unrelated



# Semantic similarity: How similar are two linguistic items?

- Defined in many levels
  - Words, word senses or concepts, phrases, paragraphs, documents
- Similarity is a specific type of relatedness
  - **Related**: topically or via relation  
*heart vs surgeon*  
*wheel vs bike*
  - **Similar**: synonyms and hyponyms  
*doctor vs surgeon*  
*bike vs bicycle*

# Semantic similarity: Numerous attempts to answer that

Allison and Dix (1986)  
Gusfield (1997)  
Wise (1996)  
Keselj et al. (2003)  
50+ Approaches from  
SemEval  
2012, 2013, 2014

Sussna (1993, 1997)  
Wu and Palmer (1994)  
Resnik (1995)  
Jiang and Conrath (1997)  
Lin (1998)  
Hirst and St-Onge (1998)  
Leacock and Chodorow (1998)  
Patwardan (2003)  
Banerjee and Pederson (2003)

Salton and McGill (1983)  
Landauer et al. (1998)  
Turney (2007)

Gabrilovich and Markovitch (2007)  
Ramage et al. (2009)  
Yeh et al. (2009)  
Radinsky et al. (2011)

**We refer to these as  
Linguistic Levels**



Sentence

Word

Sense

\*Image from D. Jurgens' NAACL 2016 tutorial.



# Semantic similarity: Numerous attempts to answer that

Allison and Dix (1986)  
Gusfield (1997)  
Wise (1996)  
Keselj et al. (2003)  
50+ Approaches from  
SemEval  
2012, 2013, 2014

**Not to mention  
word embeddings...**



Sussna (1993, 1997)  
Wu and Palmer (1994)  
Resnik (1995)  
Jiang and Conrath (1997)  
Lin (1998)  
Hirst and St-Onge (1998)  
Jeacock and Chodorow (1998)  
Patwardan (2003)  
Banerjee and Pederson (2003)

Gat

7)

Ramage et al. (2009)  
Yeh et al. (2009)  
Radinsky et al. (2011)

**We refer to these as  
Linguistic Levels**



Sentence

Word

Sense

# Semantic similarity: Why do we have so many methods?

- New resources or methods
  - Datasets reveal weakness in previous methods
  - State-of-the-art is moving target
- Task-specific similarity functions
- Performance in new tasks not satisfactory
- ➔ Semantic similarity is not the end-task
  - Pick the one which yields best results
  - Need for methods to quickly adapt similarity

# Two main sources for measuring similarity



**Massive text corpora**

**WordNet**  
A lexical database for English



**WIKTIONARY**  
*the free dictionary*



BabelNet



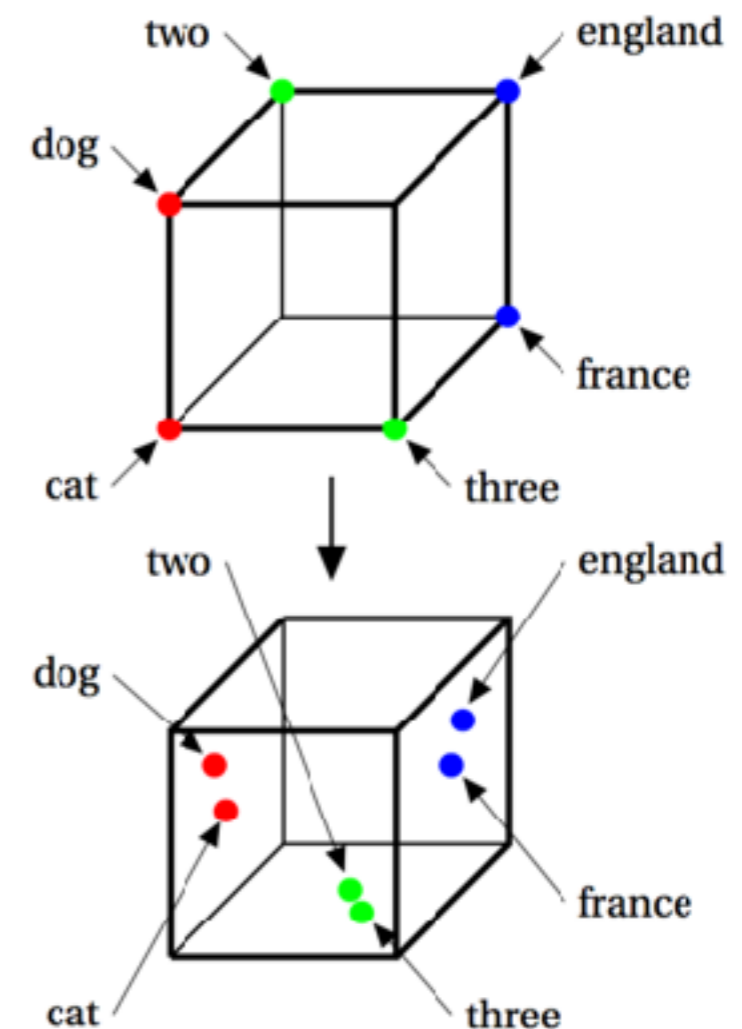
**WIKIPEDIA**  
The Free Encyclopedia

**Semantic resources and knowledge bases**

# How to Represent Word 'Meaning'?

- **Discrete:** each dimension denotes a specific linguistic item
  - Interpretable dimensions
  - High dimensionality
- **Continuous:** dimensions are not tied to explicit concepts
  - Enable comparison between represented linguistic items
  - Low dimensionality

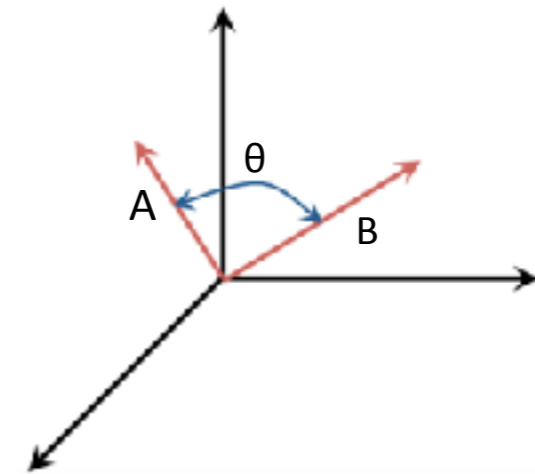
dog = [0, 0, 0, 1, 0, 0]  
cat = [0, 1, 0, 0, 0, 0]  
sim(dog, cat) = 0.0



# How to compare two linguistic items in the vector space

- Cosine of the angle  $\theta$  between A and B:

$$\cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\| \|\mathbf{B}\|} = \frac{\sum_{i=1}^n A_i B_i}{\sqrt{\sum_{i=1}^n A_i^2} \sqrt{\sum_{i=1}^n B_i^2}}$$



- Explicit models have a serious sparsity problem due to their discrete or “k-hot” vector representations

*france* = [0, 0, 0, 1, 0, 0]

*england* = [0, 1, 0, 0, 0, 0]

*france is near spain* = [1, 0, 0, 1, 1, 1]

- $\cos(\textit{france}, \textit{england}) = 0.0$
- $\cos(\textit{france}, \textit{france is near spain}) = 0.57$



# Learning Word Representations From Text

- Limitations of knowledge-based methods
  - Out-of-context despite validity of resources
  - Most lack of evaluation on practical tasks
- What if we do not know anything about words?
  - Follow the distributional hypothesis (unsupervised): “You shall know a word by the company it keeps”, [Firth 1957](#)



The value of the central **bank** increased by 10%.  
She often goes to the **bank** to withdraw cash.  
She went to the river **bank** to have picnic with her child.

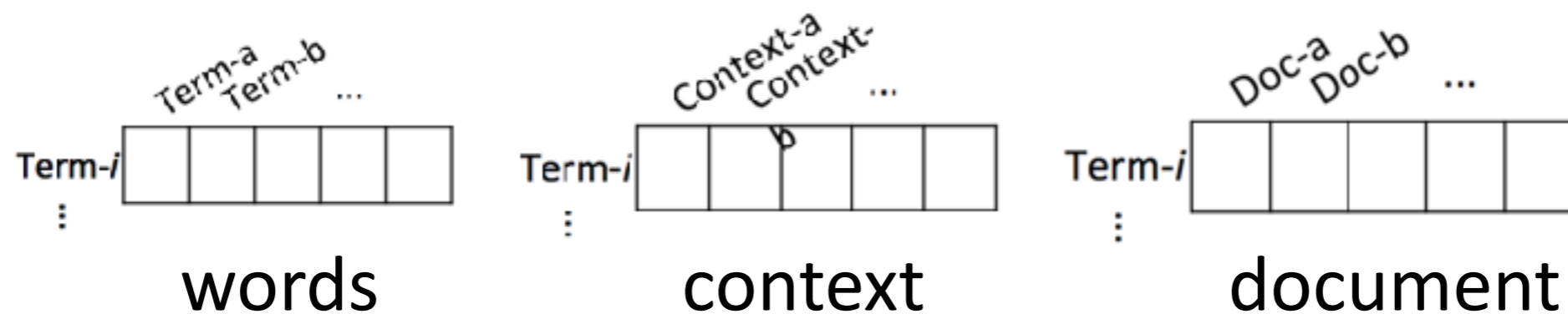
financial institution

geographical term



# Simple approach: Compute a word-in-context co-occurrence matrix

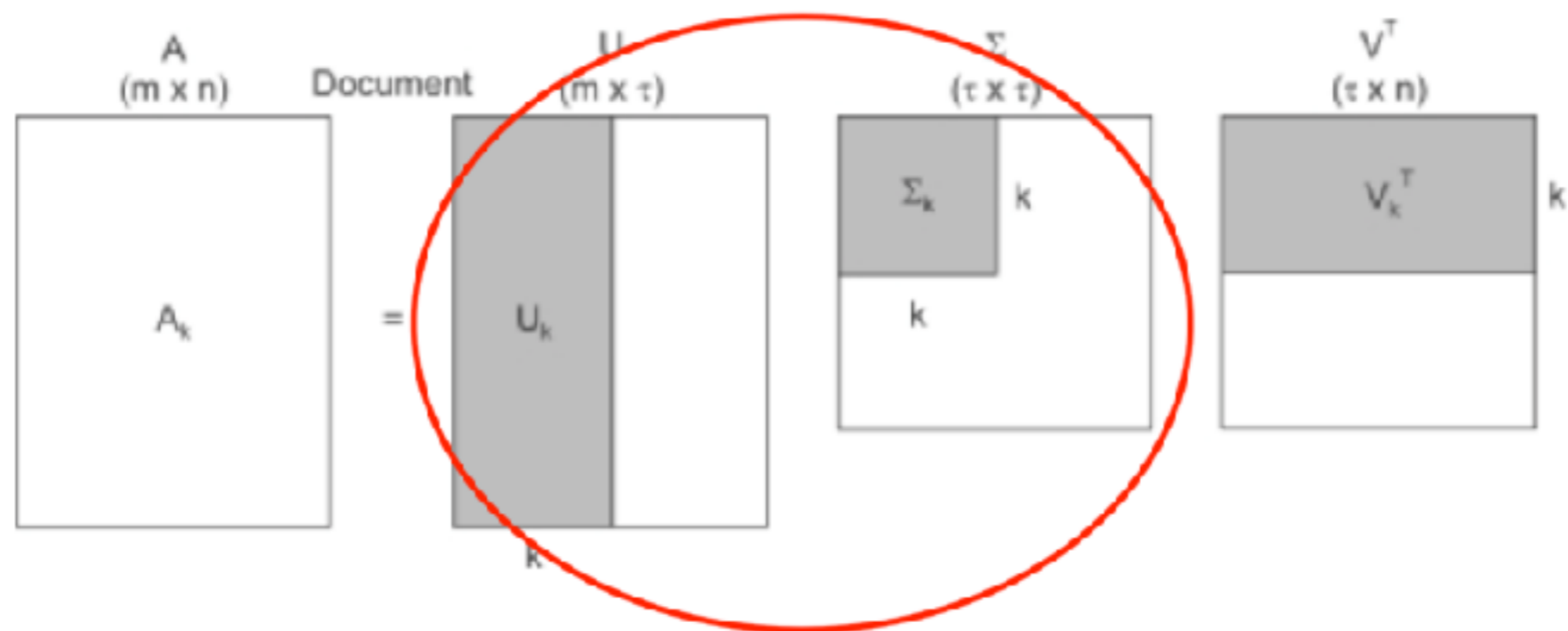
- Matrix of counts between words and contexts



- **Limitations**
  - All words have equal importance (imbalance)
  - Vectors are very high dimensional (storage issue)
  - Infrequent words have overly sparse vectors (make subsequent models less robust)

# The most standard approach: Dimensionality Reduction

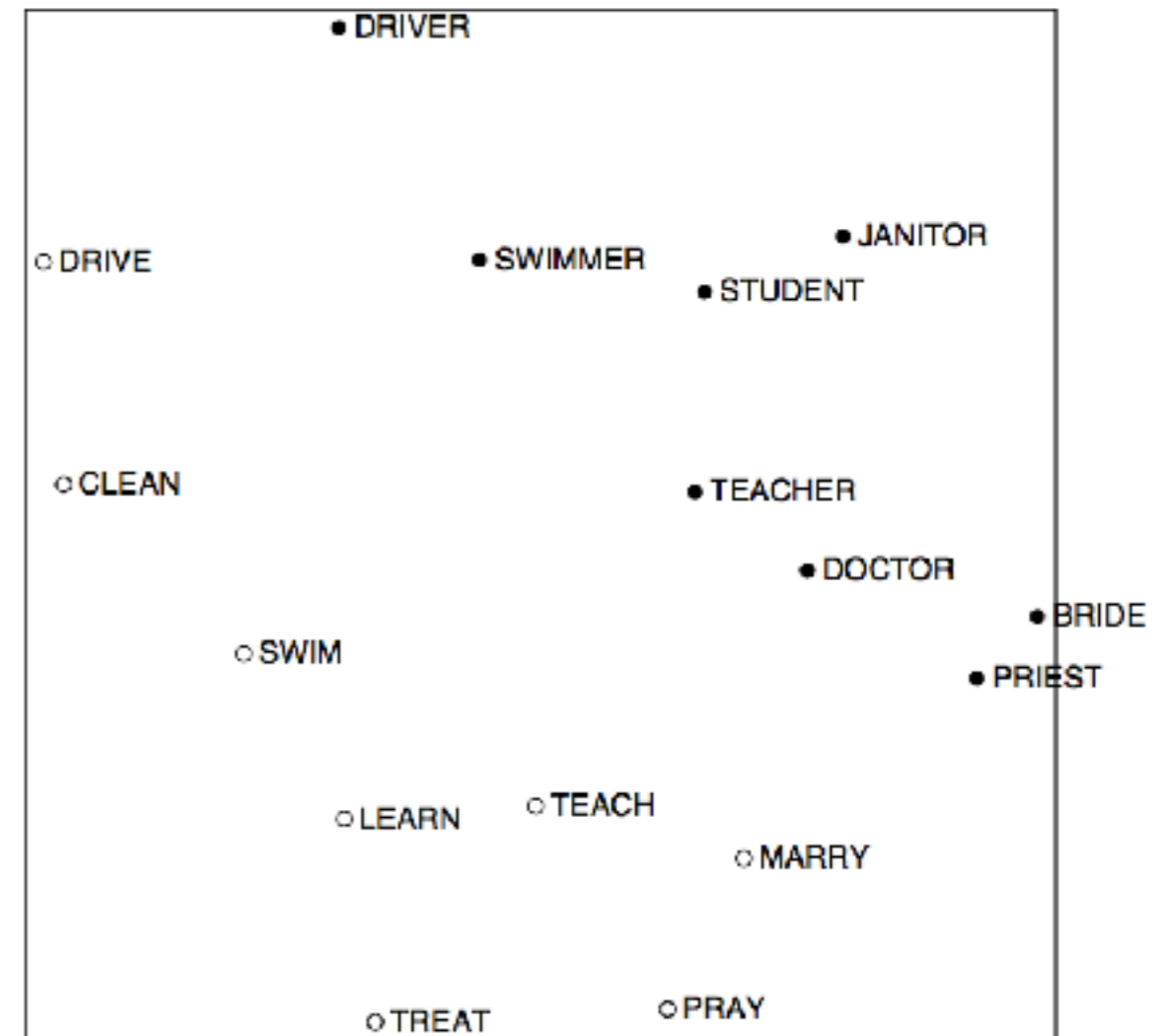
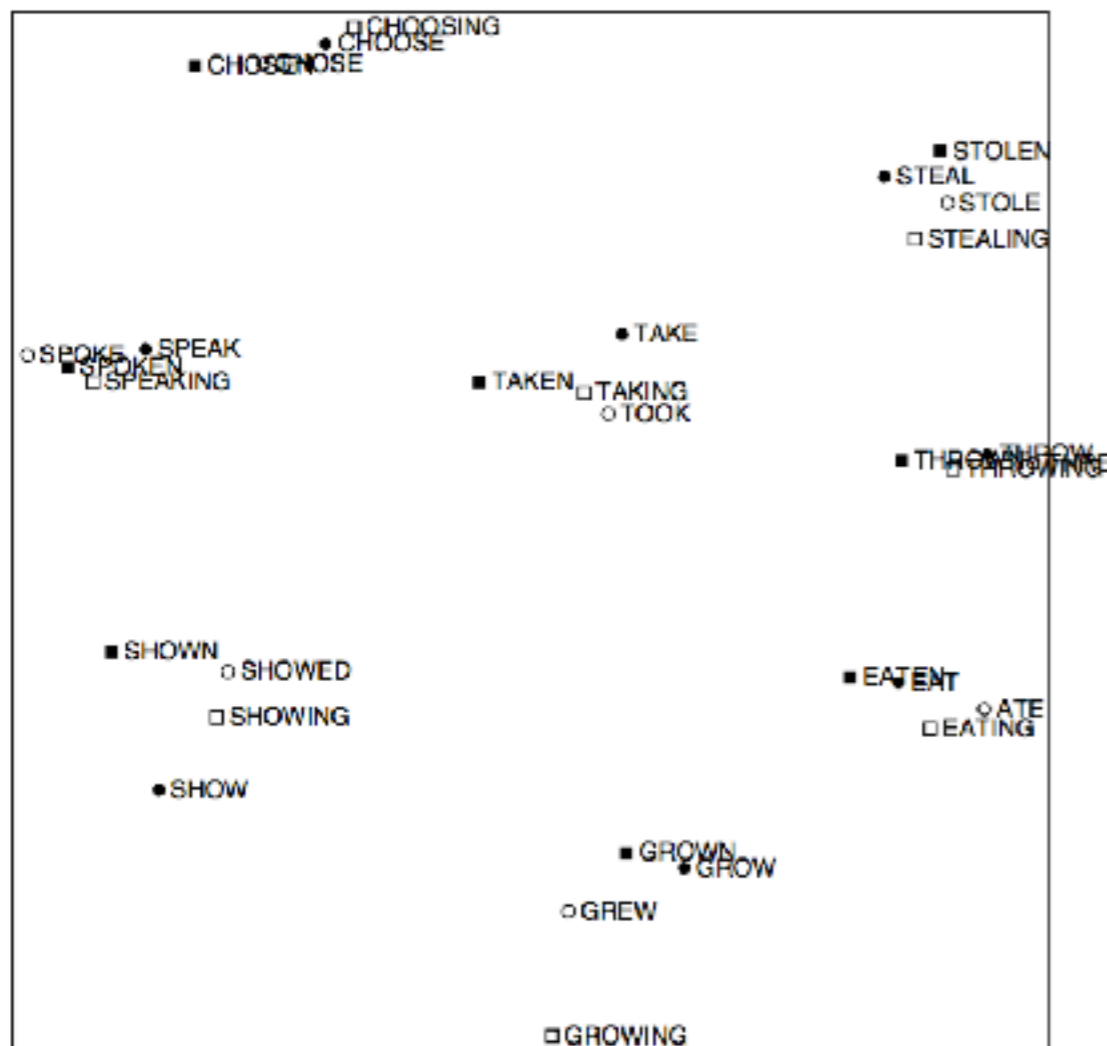
- Perform singular value decomposition (SVD) of the word co-occurrence matrix that we saw previously
  - Typically,  $U^* \Sigma$  is used as the vector space



\*Image from D. Jurgens' NAACL 2016 tutorial.

# The most standard approach: Dimensionality Reduction

- Syntactically and semantically related words cluster together



\*Plots from Rohde et al. 2005

# Dimensionality reduction with Hellinger PCA

- Perform PCA with Hellinger distance on the word co-occurrence matrix: [Lebret and Collobert 2014](#)
  - Well suited for discrete probability distributions (P, Q)

$$H(P, Q) = \frac{1}{\sqrt{2}} \sqrt{\sum_{i=1}^k (\sqrt{p_i} - \sqrt{q_i})^2},$$

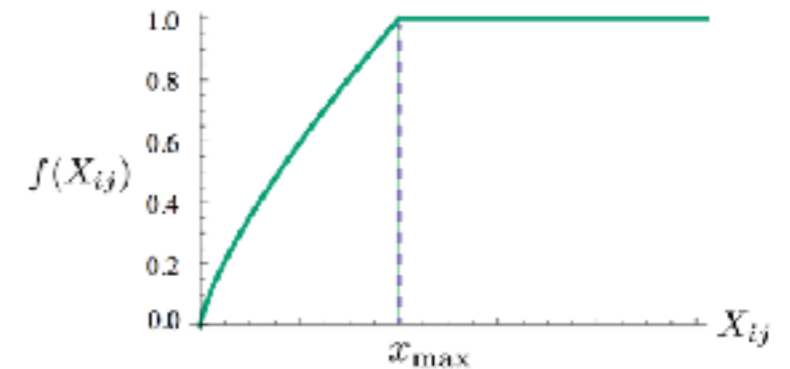
- Neural approaches are time-consuming (tuning, data)
  - Instead compute word vectors efficiently with PCA
  - Fine-tuning them on tasks; better than neural
- **Limitations:** hard to add new words, not scalable  $O(mn^2)$

<https://github.com/rlebret/hpca>

# Dimensionality reduction with weighted least squares

- Glove vectors by [Pennington et al 2014](#). Factorizes the log of the co-occurrence matrix:

$$J(\theta) = \frac{1}{2} \sum_{i,j=1}^W f(P_{ij})(u_i^T v_j - \log P_{ij})$$



- Fast training, scalable to huge corpora but still hard to incorporate new words
- Much better results than neural embedding, however under equivalent tuning it is not the case: [Levy and Goldberg 2015](#)

<http://nlp.stanford.edu/projects/glove/>

# Dimensionality reduction with neural networks

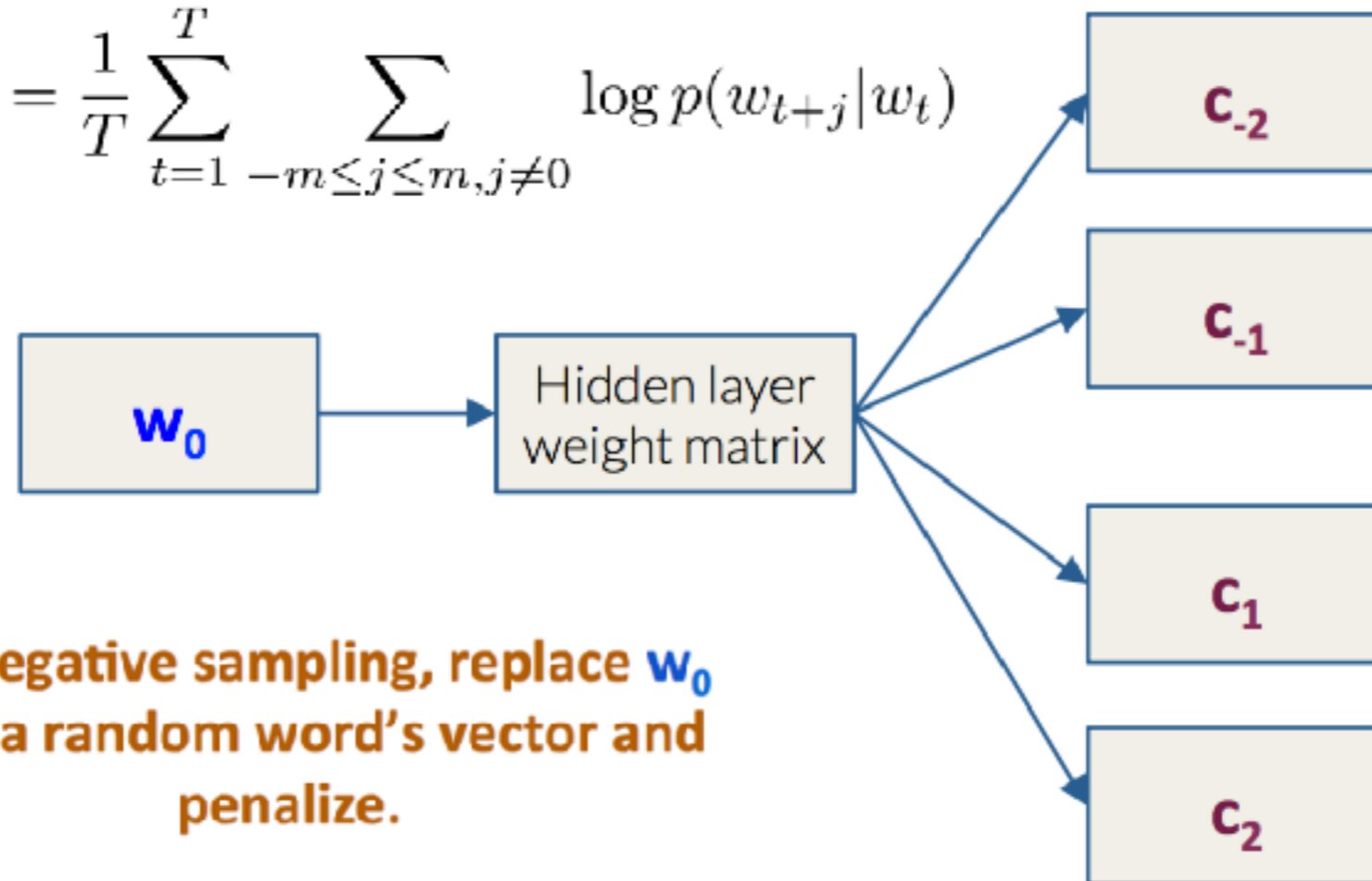
- The main idea is to directly learn low-dimensional word representations from data
  - Learning representations: [Rumelhart et al 1986](#)
  - Neural probabilistic language model: [Bengio et al 2003](#)
  - NLP (almost) from scratch: [Collobert and Weston 2008](#)
- Recent methods are faster and more simple
  - Continuous Bag-Of-Words (CBOW)
  - Skip-gram with Negative Sampling (SGNS)
  - word2vec toolkit: [Mikolov et al. 2013](#)



# word2vec: Skip-gram with negative sampling (SGNS)

- Given the middle word predict surrounding ones in a fixed window of words (maximize log likelihood)

$$J(\theta) = \frac{1}{T} \sum_{t=1}^T \sum_{-m \leq j \leq m, j \neq 0} \log p(w_{t+j} | w_t)$$



**For negative sampling, replace  $w_0$  with a random word's vector and penalize.**

# word2vec: Skip-gram with negative sampling (SGNS)

- How is the  $P(w_t|h)$  probability computed?

$$\begin{aligned} P(w_t|h) &= \text{softmax}(\text{score}(w_t, h)) \\ &= \frac{\exp\{\text{score}(w_t, h)\}}{\sum_{\text{Word } w' \text{ in Vocab}} \exp\{\text{score}(w', h)\}} \end{aligned}$$

- Denominator is very costly for big vocabulary!
- Instead it uses a more scalable objective,  $\log Q_\theta$  is a binary logistic regression of word  $w$  and history  $h$ :

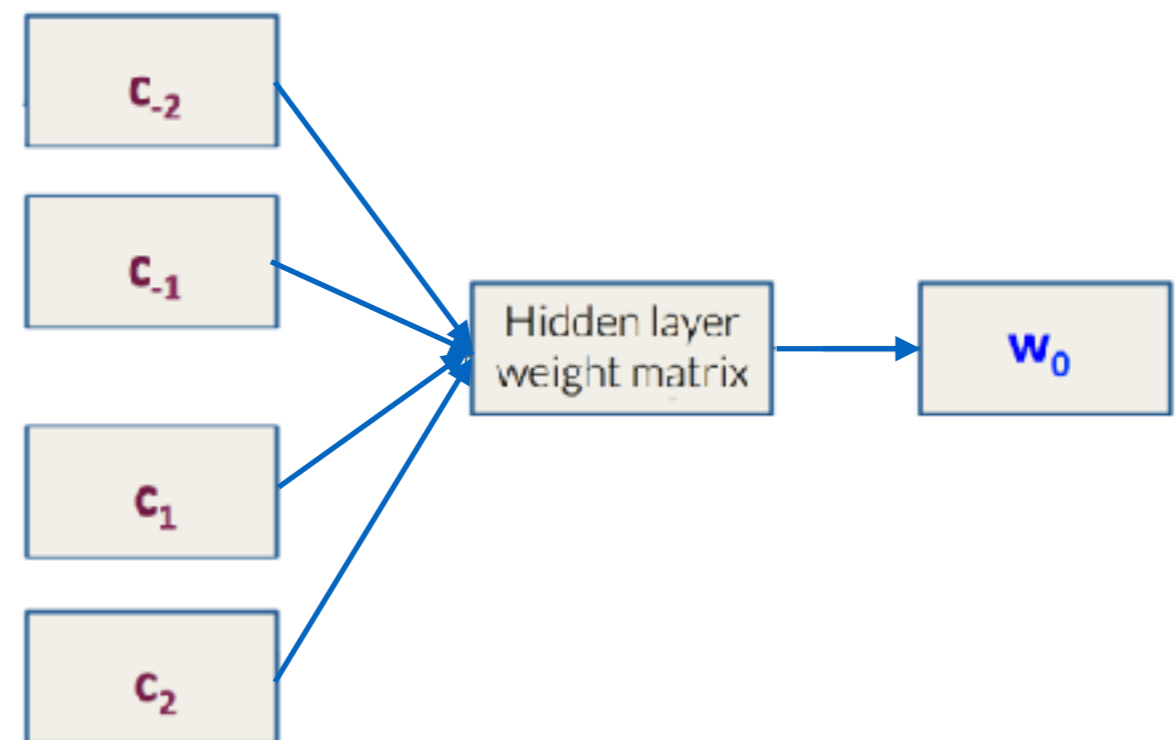
$$J_{\text{NEG}} = \log Q_\theta(D = 1|w_t, h) + k \mathbb{E}_{\tilde{w} \sim P_{\text{noise}}} [\log Q_\theta(D = 0|\tilde{w}, h)]$$

# word2vec: Continuous Bag-Of-Words with negative sampling (CBOW)

- More efficient but the ordering information of the words does not influence the projection

- Factorizes a PMI word-context matrix: [Levy and Goldberg 2014](#)

- Builds upon existing methods (new decomp.)
- Improvements on a variety of intrinsic tasks such as relatedness, categorization and analogy: [Baroni et al 2014](#), [Schnabel et al 2015](#)

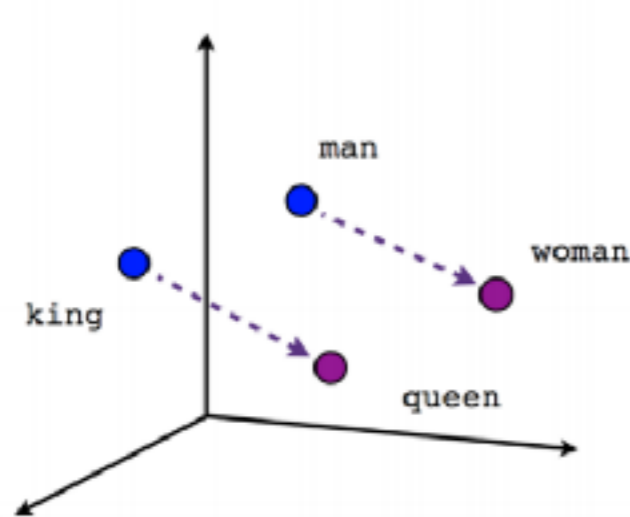


	RG	WordSim	MEN	TOEFL
PMI+SVD	.70	.70	.72	.76
word2vec	.83	.78	.80	.86

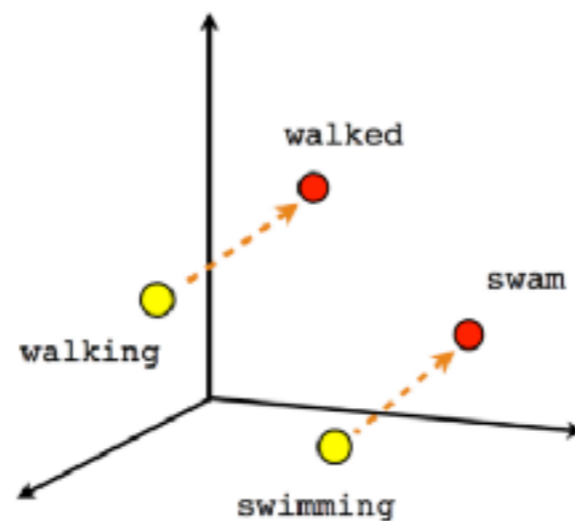
# Distributed representations: Encoded properties

- Encodes general-purpose relations between words: present—past tense, singular—plural, male—female, capital—country
- Analogy between words can be efficiently computed using basic arithmetic operations between vectors (+, -)

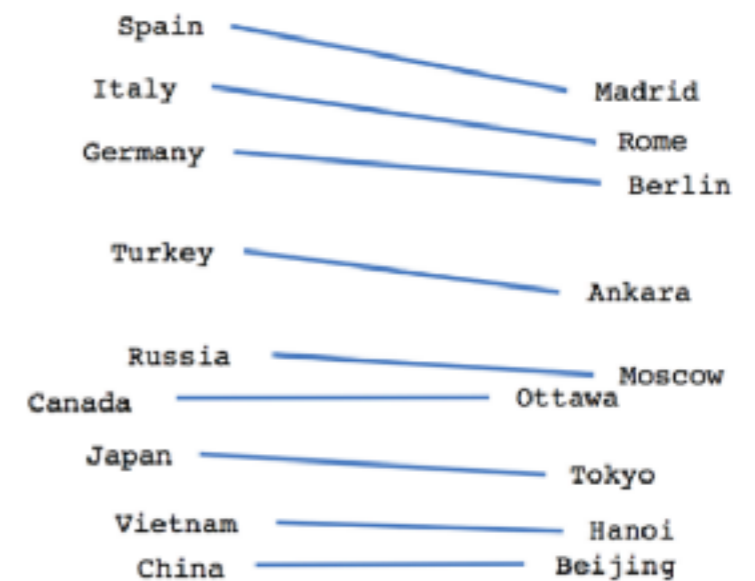
king - man + woman  $\approx$  queen



Male-Female



Verb tense



Country-Capital

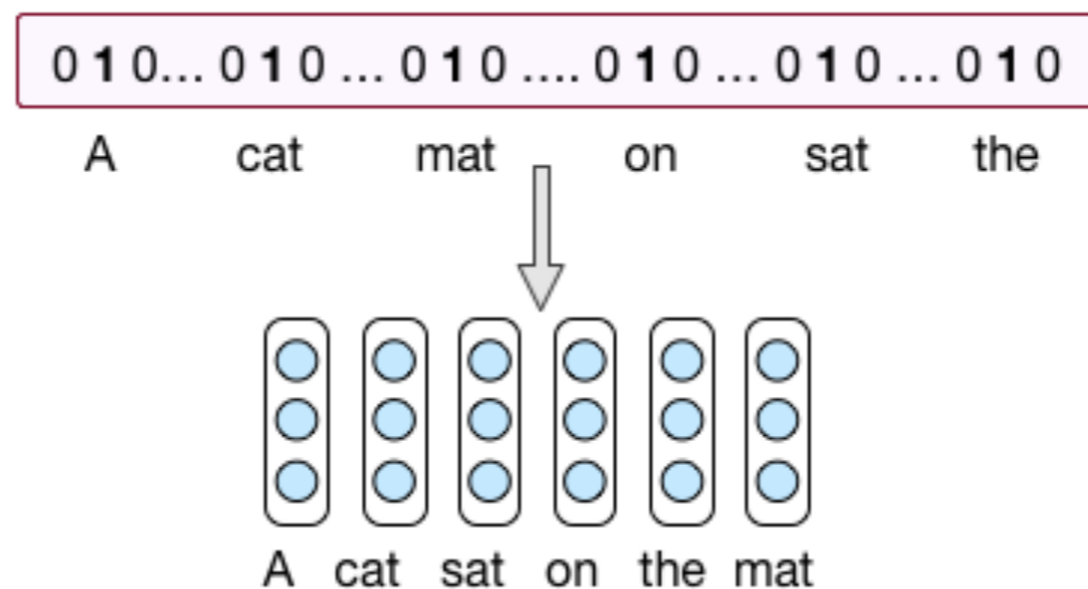
# Summary: Learning word representations

- Neural versus count-based methods
  - neural ones implicitly do SVD over a PMI matrix
  - similar to count-based when using the same tricks
- Neural methods appear to have the edge (word2vec)
  - efficient and scalable objective + toolkit
  - intuitive formulation (=predict words in context)
- ➔ **Several extensions**
  - Dependency-based embeddings: [Levy and Goldberg 2014](#)
  - Retrofitted-to-lexicons embeddings: [Faruqui et al. 2014](#)
  - Sense-aware embeddings: [Li and Jurafsky 2015](#)
  - Visually-grounded embeddings: [Lazaridou et al. 2015](#)
  - Multilingual embeddings: [Gouws et al 2015](#)

# Summary: Learning word representations

## How can we benefit from them?

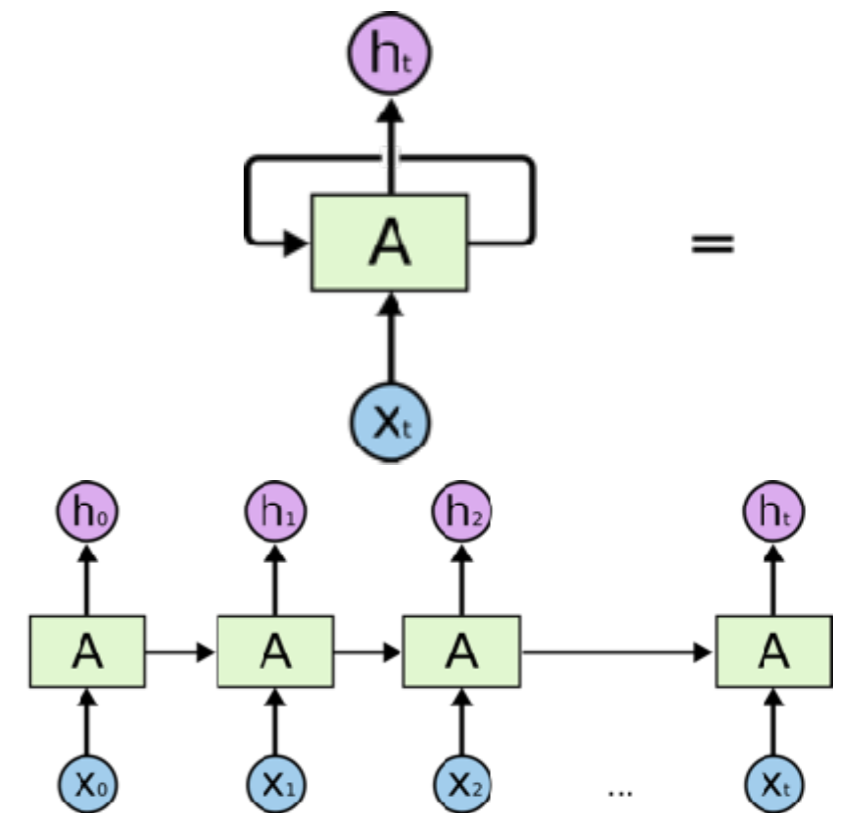
- study linguistic properties of words
- inject general knowledge on downstream tasks
- transfer knowledge across languages or modalities
- representations of word sequences





# Outline of the talk

1. Introduction and Motivation
2. Word Representation Learning
  - Semantic similarity
  - Traditional and recent approaches
  - Intrinsic and extrinsic evaluation
3. Word Sequence Modeling
  - Essentials: RNNs, Attention, DL tricks
  - Text Classification
  - Machine Translation
4. Conclusion and Discussion



\* Figure from Colah's blog, 2015.

# Language Modeling

- Computes the joint probability of a sequence of words by employing the chain rule (“How likely is a text”):

$$p(w_1, w_2, \dots, w_t) = p(w_1)p(w_2|w_1)p(w_3|w_2, w_1)\dots p(w_t|w_{t-1}, w_{t-2}, \dots)$$

- Given the observed text how likely is the new utterance?

$$p(w_t | w_{t-1}, \dots, w_1)$$

- Hence, we can compare orderings (translation)

$$p(\text{he likes apples}) > p(\text{apples likes he})$$

or word choice (speech recognition)

$$p(\text{he likes apples}) > p(\text{he licks apples})$$

- ➔ Exact decomposition allows to learn complex distributions
- ➔ Many NLP tasks can be structured as (conditional) language model

Evaluation

$$H(w_1^N) = -\frac{1}{N} \log_2 p(w_1^N)$$

$$\text{perplexity}(w_1^N) = 2^{H(w_1^N)}$$

# Language Modeling: Markov Models

- N-gram models: history of observed words is approximated with just the previous n words (Markov model):
  - hard to capture long-term dependencies (bounded memory)
  - does not leverage word semantics and relationships

$$p(w_3|w_1, w_2) = \frac{\text{count}(w_1, w_2, w_3)}{\text{count}(w_1, w_2)}$$

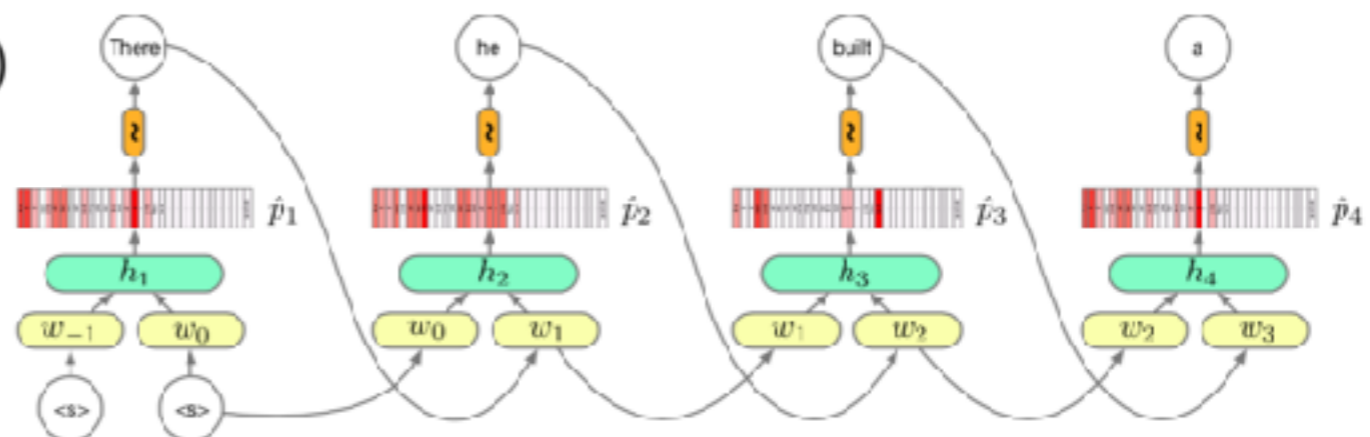
- Neural n-gram models: embed the same fixed n-gram history in a continuous space (still Markov model)
  - captures better correlations + smaller memory footprint

$$h_n = g(V[w_{n-1}; w_{n-2}] + c)$$

$$\hat{p}_n = \text{softmax}(Wh_n + b)$$

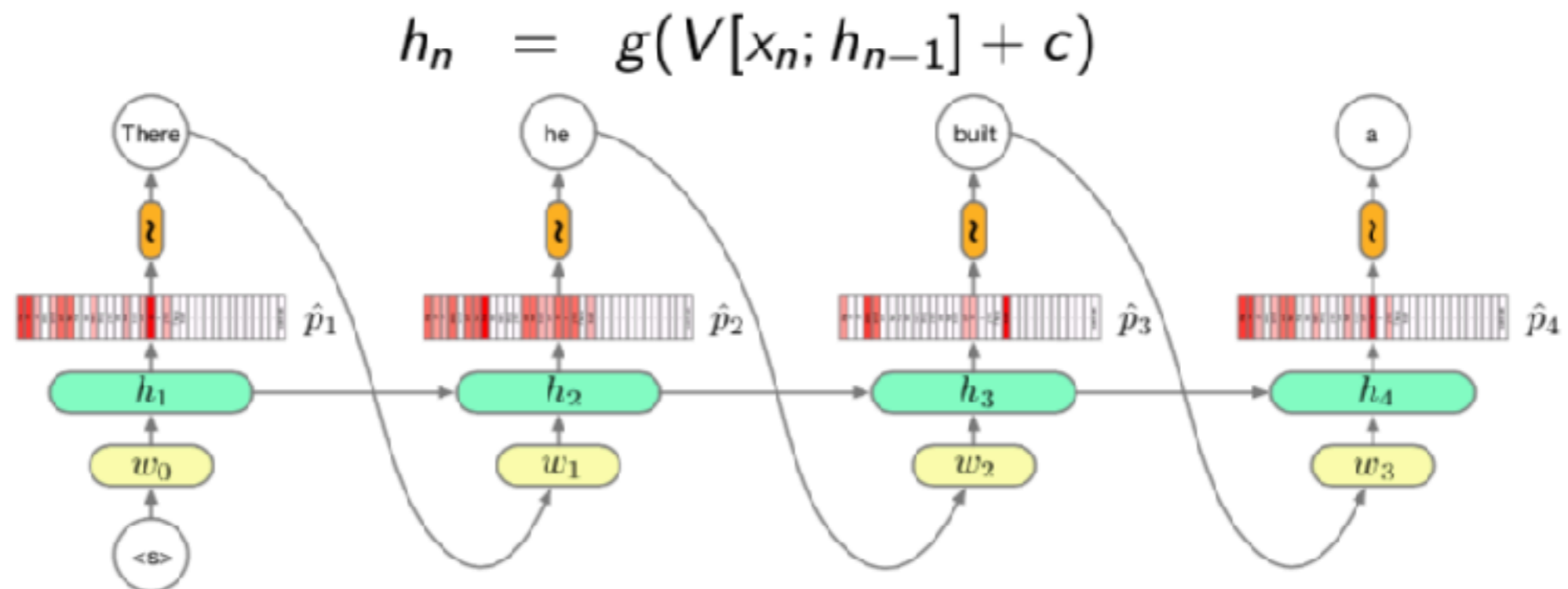
- trained with MLE

$$\mathcal{F} = -\frac{1}{N} \sum_n \text{cost}_n(w_n, \hat{p}_n)$$



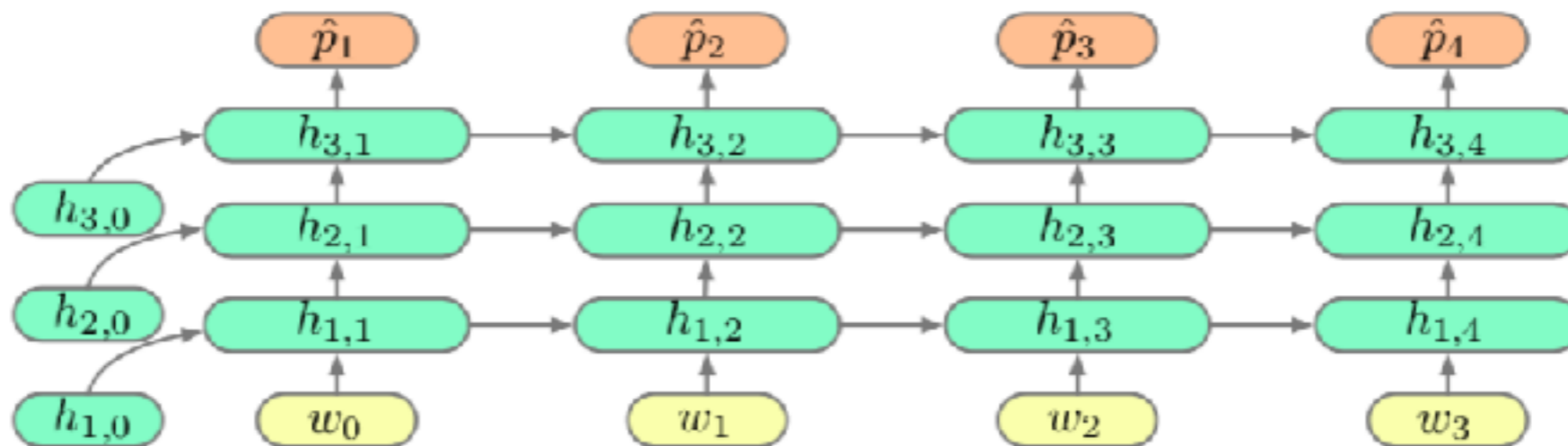
# Language Modeling: Recurrent Neural Networks (RNN)

- With RNN LMs we drop the fixed n-gram history and compress the entire history in a fixed length vector
  - long range correlations are captured — in theory
  - can represent unbounded dependencies
  - but, they are **hard** to learn (vanishing gradient)



# Language Modeling: Deep RNNs

- Increasing the size of the hidden layer results in a quadratic increase in the model size and computation
- Stacking multiple RNNs increases the **memory capacity** and **representational ability** with linear scaling
- We can also increase depth in the time dimension



# Scaling: Large Vocabularies

- Much of the computational cost comes from the classification layer because its parameters depend on the size of the vocabulary:

$$\hat{p}_n = \text{softmax}(Wh_n + b)$$

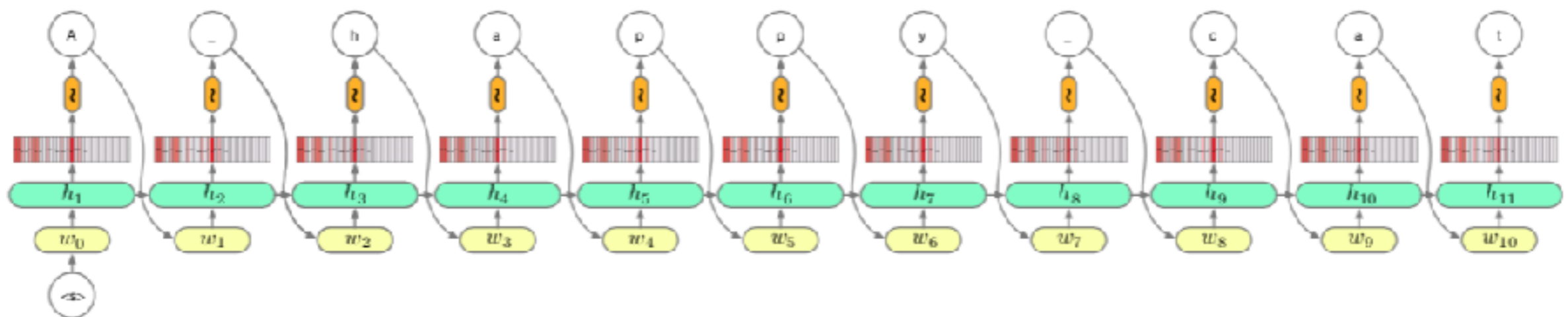
- Several solutions exist
  - Short-lists: use most frequent words + n-gram LM for the rest
  - Local short-lists: subsets of vocabulary specific to data segments
  - Gradient approximations: use Noise Contrastive Estimation (NCE) i.e. learning a binary classifier to distinguish between data samples from  $k$  samples from a noise distribution:

$$p(\text{Data} = 1 | \hat{p}_n) = \frac{\hat{p}_n}{\hat{p}_n + k p_{\text{noise}}(w_n)}$$



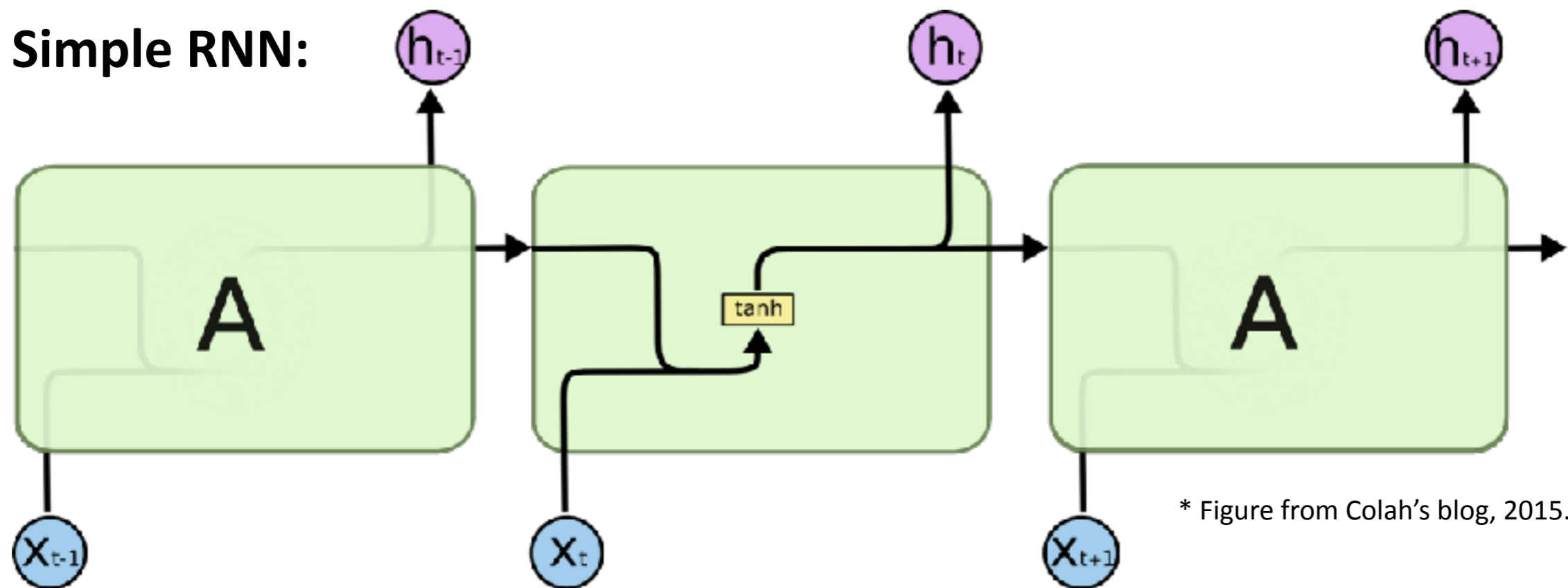
# Scaling: Large Vocabularies

- Changing the input granularity and model text at the morpheme or character level
  - Much smaller softmax but longer dependencies
  - It captures morphological properties of words
  - Byte-Pair Encoding method is most common for neural MT ([Sennrich et al 2015](#))



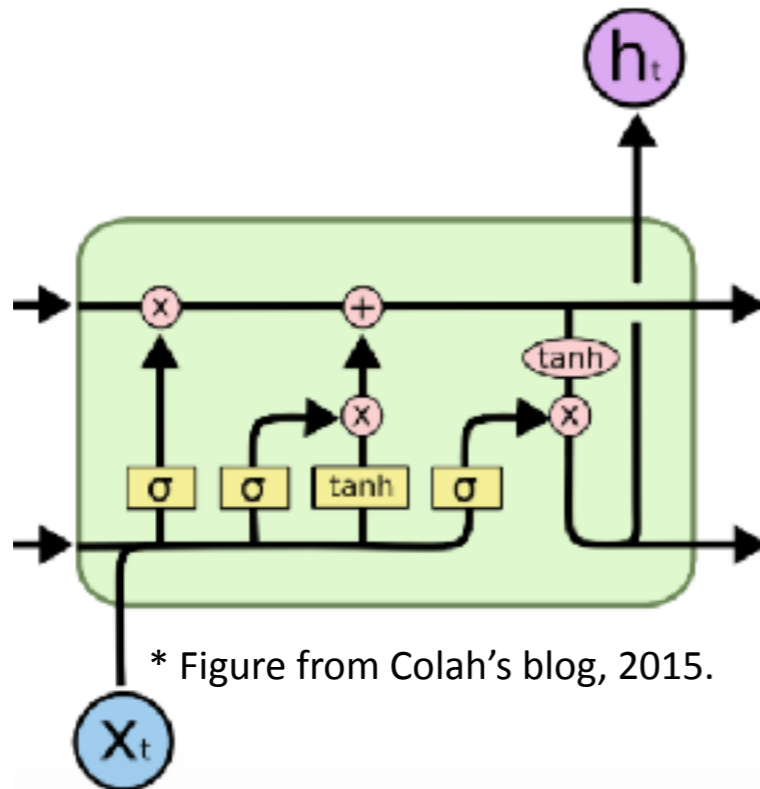
# Long Short Term Memory (LSTM)

- Long-short term memory nets are able to learn long-term dependencies: [Hochreiter and Schmidhuber 1997](#)



# Long Short Term Memory (LSTM)

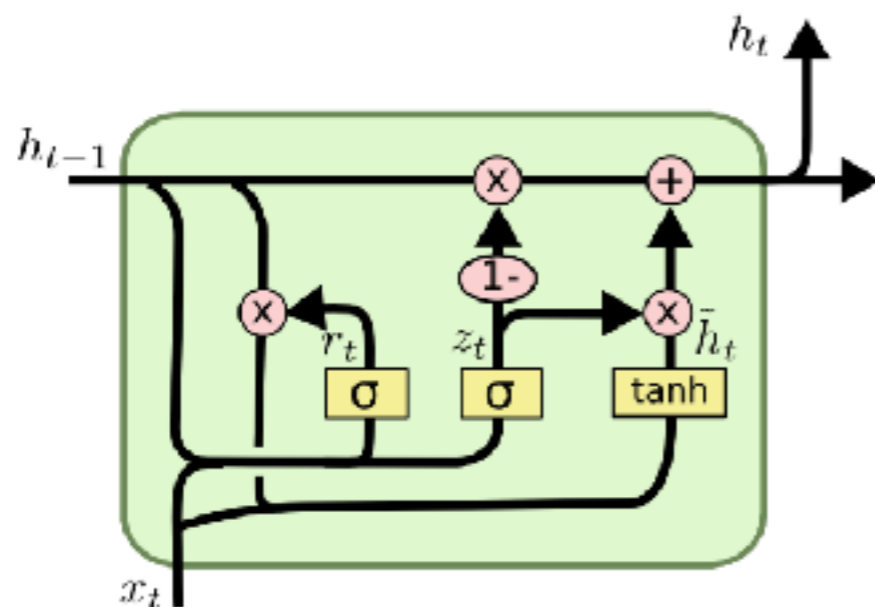
- Long-short term memory nets are able to learn long-term dependencies: [Hochreiter and Schmidhuber 1997](#)
  - Ability to remove or add information to the cell state regulated by “gates” (avoids grad. vanishing)



- Input gate (current cell matters)  $i_t = \sigma(W^{(i)}x_t + U^{(i)}h_{t-1})$
  - Forget (gate 0, forget past)  $f_t = \sigma(W^{(f)}x_t + U^{(f)}h_{t-1})$
  - Output (how much cell is exposed)  $o_t = \sigma(W^{(o)}x_t + U^{(o)}h_{t-1})$
  - New memory cell  $\tilde{c}_t = \tanh(W^{(c)}x_t + U^{(c)}h_{t-1})$
- Final memory cell:  $c_t = f_t \circ c_{t-1} + i_t \circ \tilde{c}_t$
- Final hidden state:  $h_t = o_t \circ \tanh(c_t)$

# Gated Recurrent Unit (GRU)

- Gated RNN by [Chung et al, 2014](#) combines the forget and input gates into a single “update gate”
  - keep memories to capture long-term dependencies
  - allow error messages to flow at different strengths



\* Figure from Colah's blog, 2015.

$$z_t = \sigma(W_z \cdot [h_{t-1}, x_t])$$

$$r_t = \sigma(W_r \cdot [h_{t-1}, x_t])$$

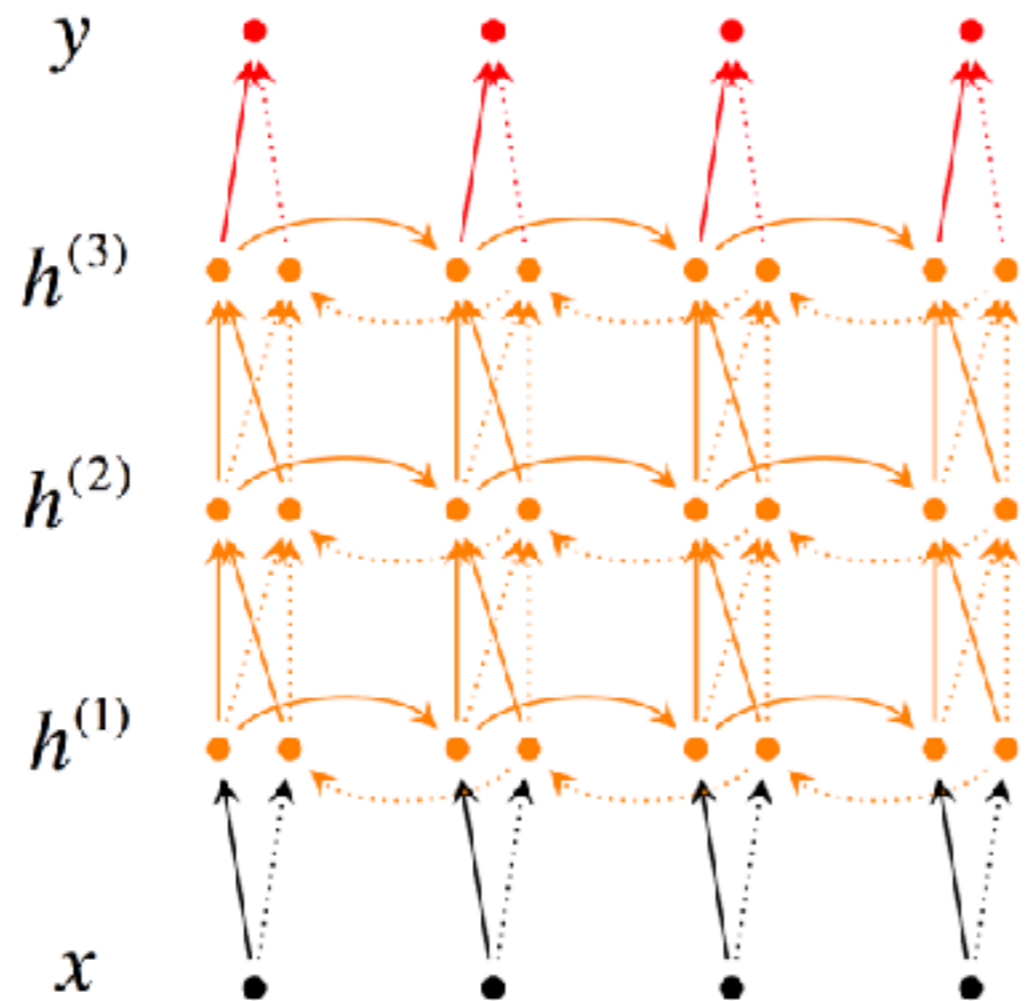
$$\tilde{h}_t = \tanh(W \cdot [r_t * h_{t-1}, x_t])$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

$z_t$ : update gate —  $r_t$ : reset gate —  $h_t$ : regular RNN update

# Deep Bidirectional Models

- Here RNN but it applies to LSTMs and GRUs too



$$\vec{h}_t^{(i)} = f(\vec{W}^{(i)} h_t^{(i-1)} + \vec{V}^{(i)} \vec{h}_{t-1}^{(i)} + \vec{b}^{(i)})$$

$$\overleftarrow{h}_t^{(i)} = f(\overleftarrow{W}^{(i)} h_t^{(i-1)} + \overleftarrow{V}^{(i)} \overleftarrow{h}_{t+1}^{(i)} + \overleftarrow{b}^{(i)})$$

$$y_t = g(U[\vec{h}_t^{(L)} ; \overleftarrow{h}_t^{(L)}] + c)$$

(Irsoy and Cardie, 2014)

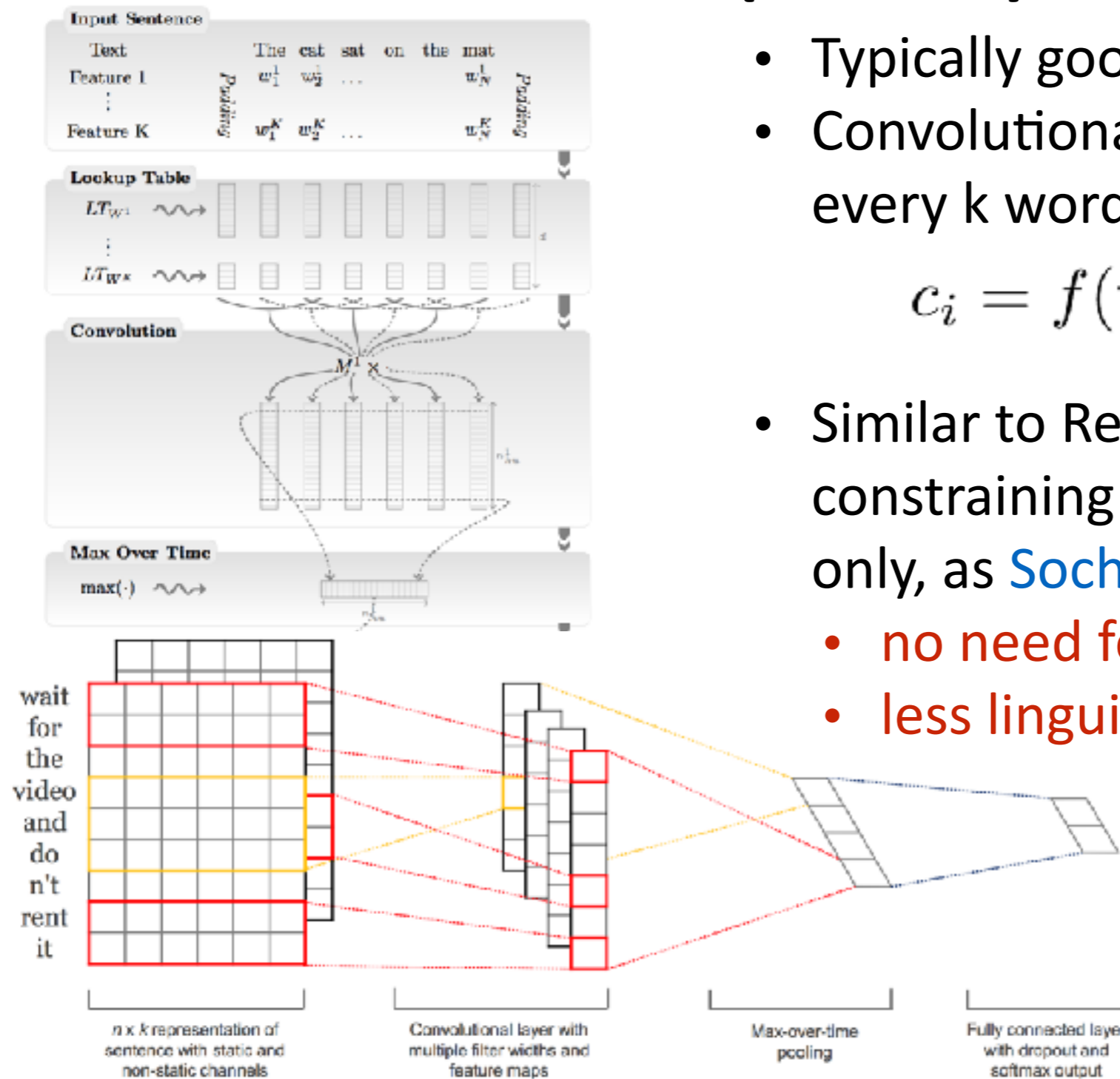
Each memory layer passes an intermediate sequential representation to the next.

# Convolutional Neural Network (CNN)

- Typically good for images
- Convolutional filter(s) is (are) applied every k words:

$$c_i = f(\mathbf{w}^T \mathbf{x}_{i:i+h-1} + b)$$

- Similar to Recursive NNs but without constraining to grammatical phrases only, as [Socher et al., 2011](#)
  - no need for a parser (!)
  - less linguistically motivated ?



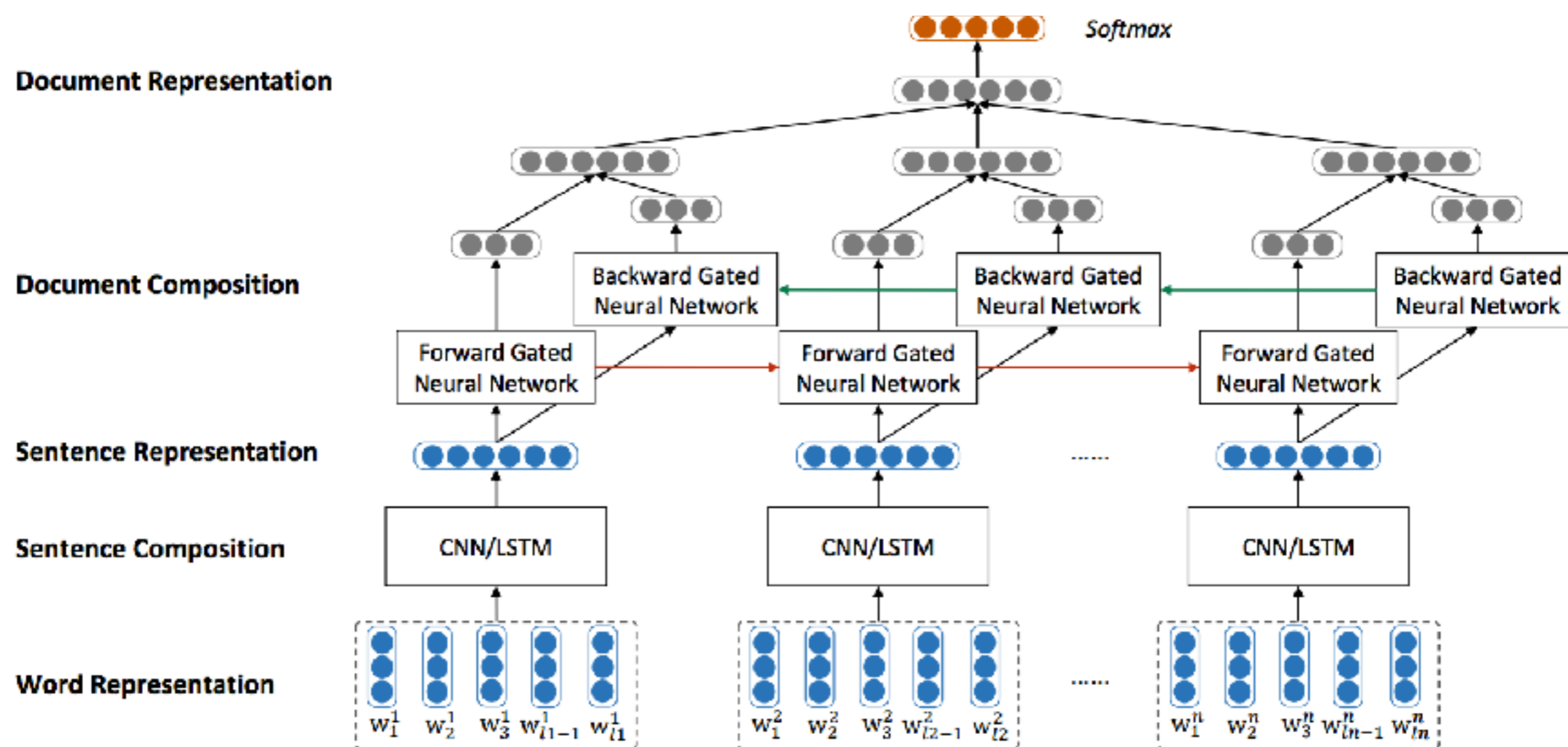
(Collobert et al., 2011)

(Kim, 2014)



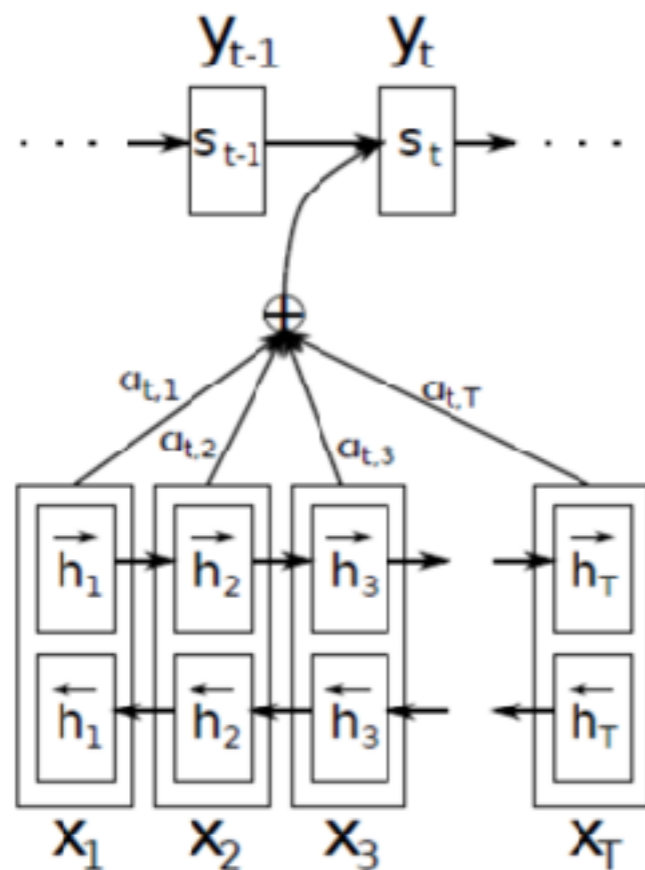
# Hierarchical Models

- Word-level and sentence-level abstractions



(Tang et al., 2015)

# Attention Mechanism: Machine Translation



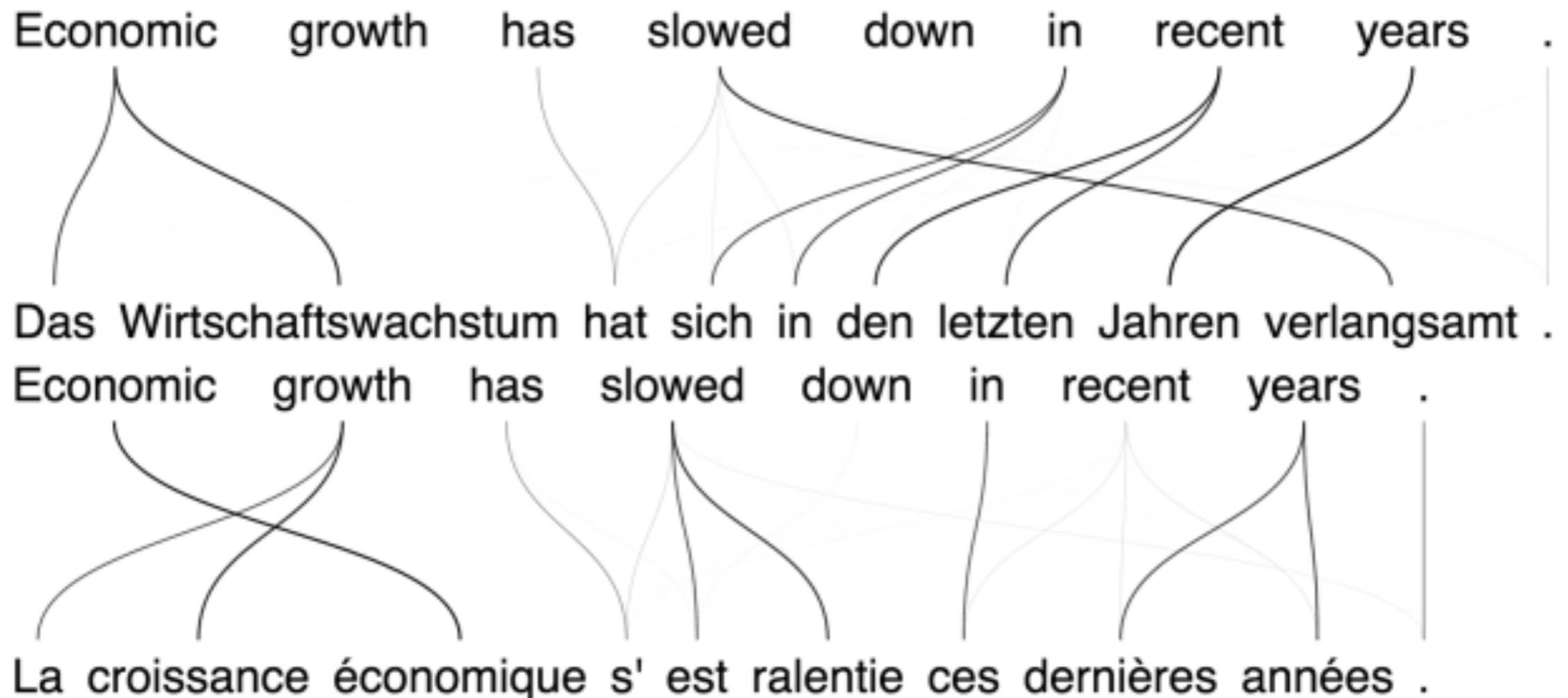
- Can we compress all the needed information in the last encoder state? **Idea:** use all the hidden states!
  - length proportional to sentence length
  - weighted average of all hidden states
- Learns to assign a relevance to each input position given current encoder state and the previous decoder state  $s_i = f(s_{i-1}, y_{i-1}, c_i)$ .
  - soft bilingual alignment model

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{T_x} \exp(e_{ik})}, \quad e_{ij} = a(s_{i-1}, h_j)$$

$$c_i = \sum_{j=1}^{T_x} \alpha_{ij} h_j.$$

(Bahdanau et al., 2015)

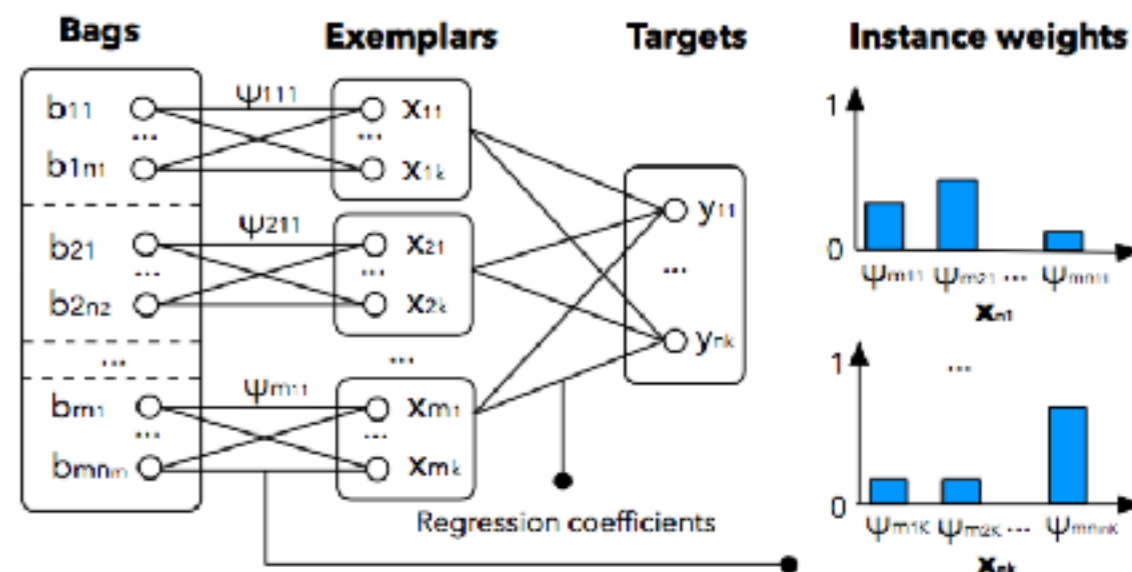
# Attention Mechanism: Machine Translation



(Bahdanau et al., 2015)

# Attention Mechanism: Document Classification

- Operates on input word sequence or intermediate hidden states
- Learns to focus on relevant parts of the input with respect to each target label
  - soft summarization model
- Can be applied at multiple language levels (Yang et al, 2016)



$$\sigma(B_i, O) = P(\psi = y_i | B_i) = \frac{\exp(O^T B_i)}{\sum_{k=1}^{n_i} \exp(O^T B_{ik})}$$

$$O, \Phi = \arg \min_{O, \Phi} \sum_{i=1}^m (y_i - \Phi^T (B_i \cdot \sigma(B_i, O)))^2 + \Omega(\Phi, O)$$

(Pappas and Popescu-Belis, 2014 & 2017)

# Hierarchical attention networks

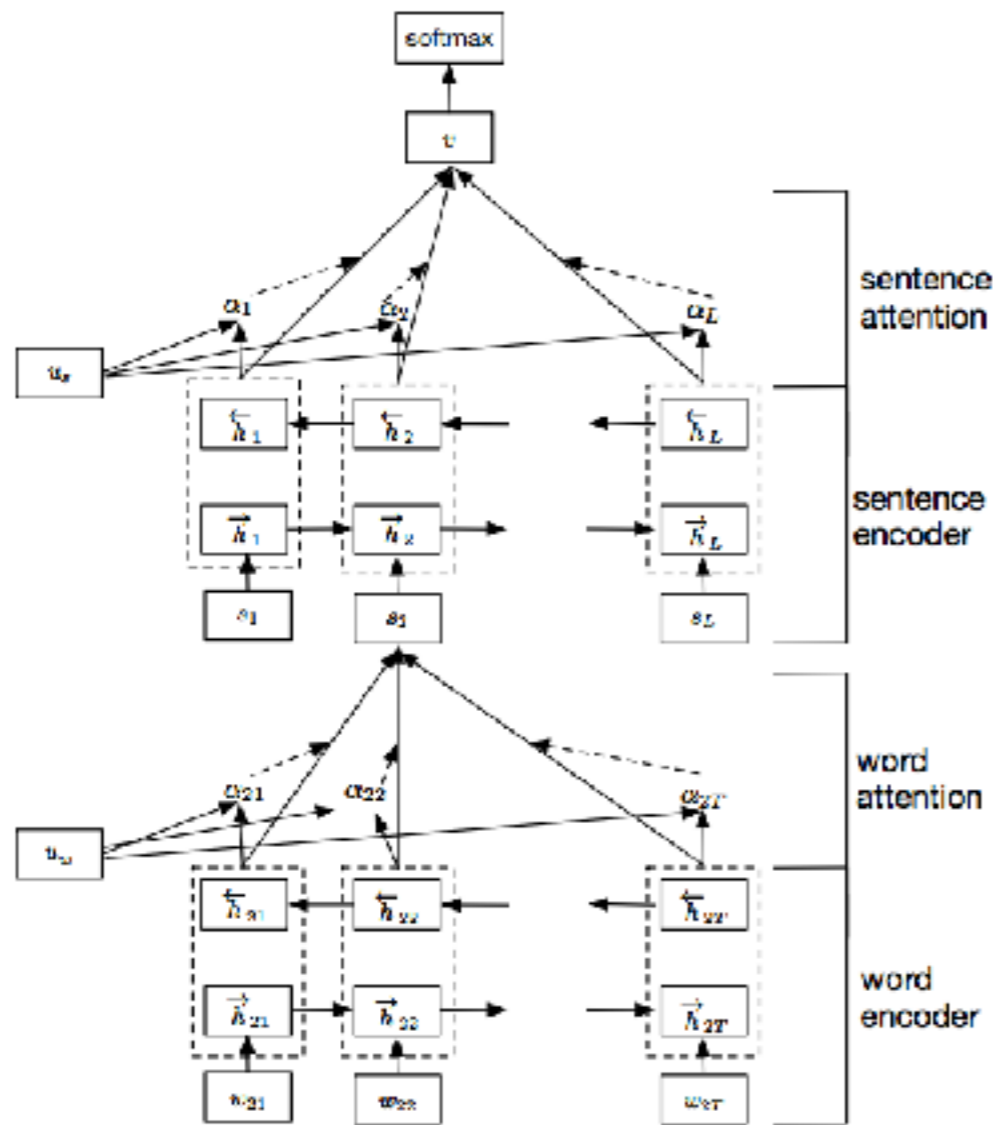


Figure 2: Hierarchical Attention Network.

- Very similar hierarchical structure as [Tang et al., 2015](#) except average pooling
- attention mechanism at the word and document levels

$$u_{it} = \tanh(W_w h_{it} + b_w)$$

$$\alpha_{it} = \frac{\exp(u_{it}^\top u_w)}{\sum_t \exp(u_{it}^\top u_w)}$$

$$s_i = \sum_t \alpha_{it} h_{it}.$$

(Yang et al., 2016)

# Attention Mechanism: Sentiment Classification

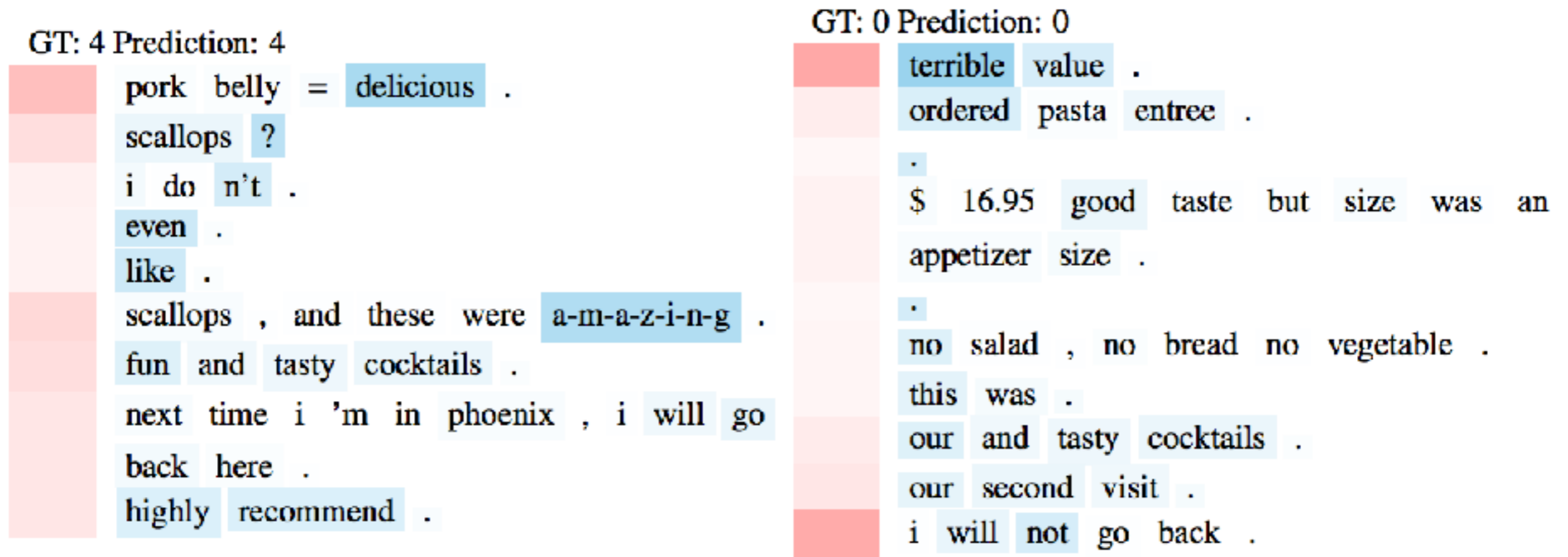


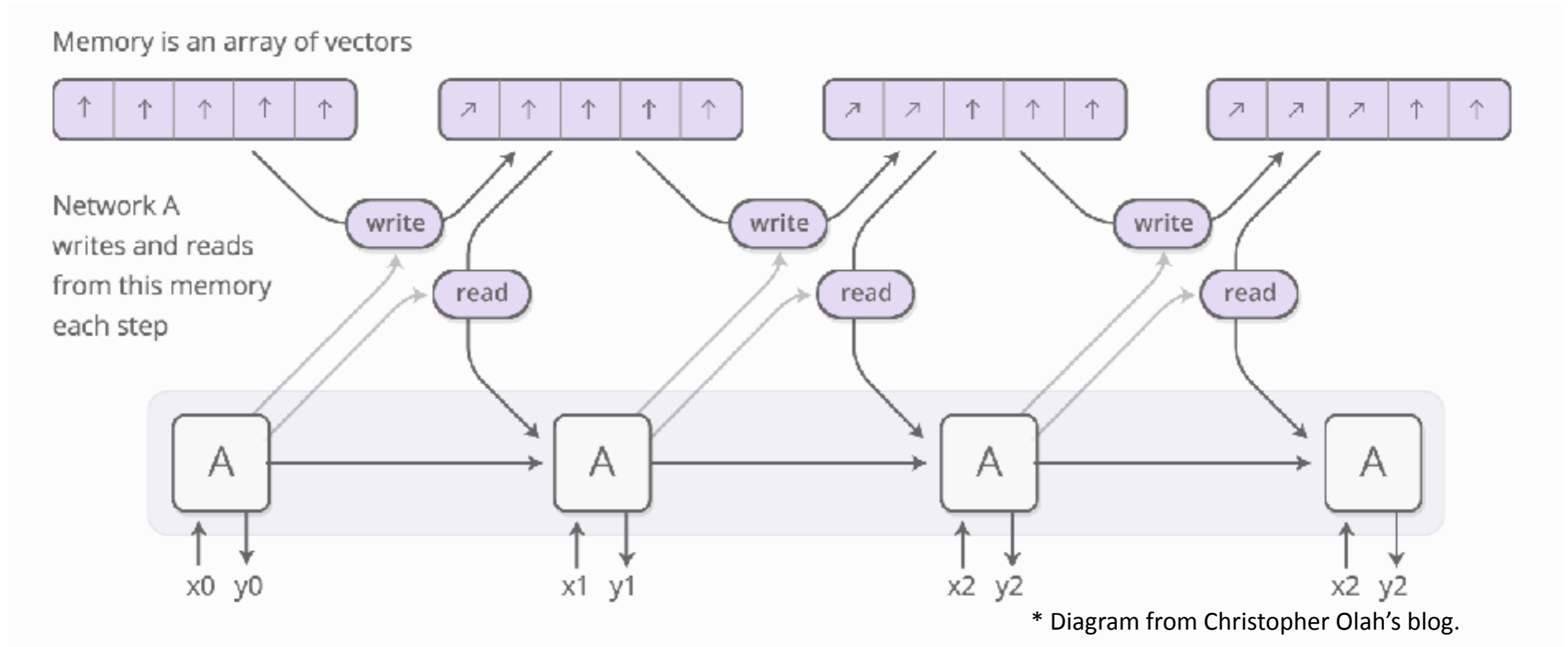
Figure 5: Documents from Yelp 2013. Label 4 means star 5, label 0 means star 1.

(Yang et al, 2016)



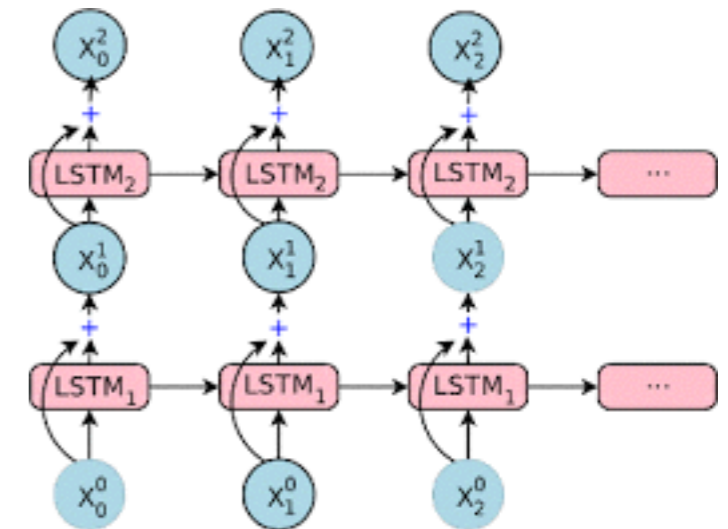
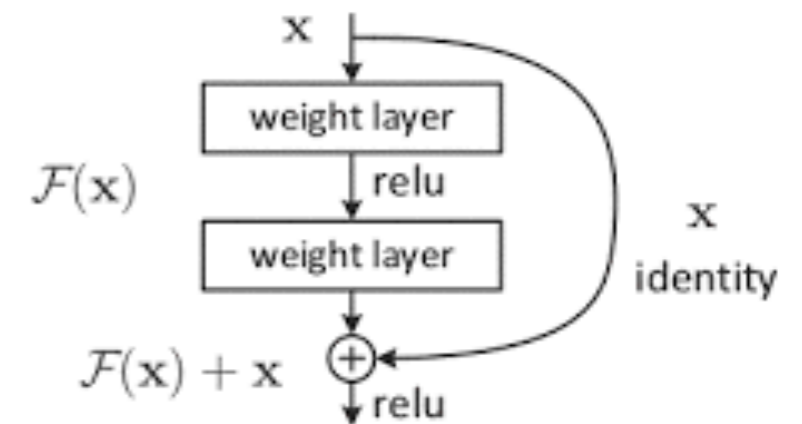
# Memory mechanism: Neural Turing Machines or Memory Networks

- Combination of recurrent network with external memory bank: [Graves et al. 2014](#), [Weston et.al 2014](#)



# Residual connections

- Residual learning allows information to flow more easily by adding the input of a layer  $F(x)$  to its output i.e.  $F(x) + x$
- It's typically used for making connections from one layer to another
  - This improves the training and avoids the vanishing gradient problem
  - Layer is ignored if not beneficial



# Other DL tricks

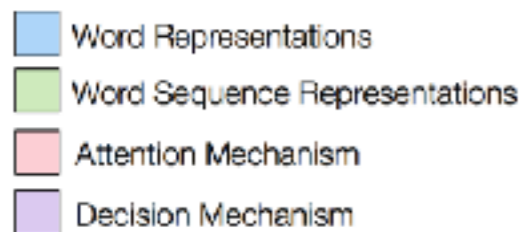
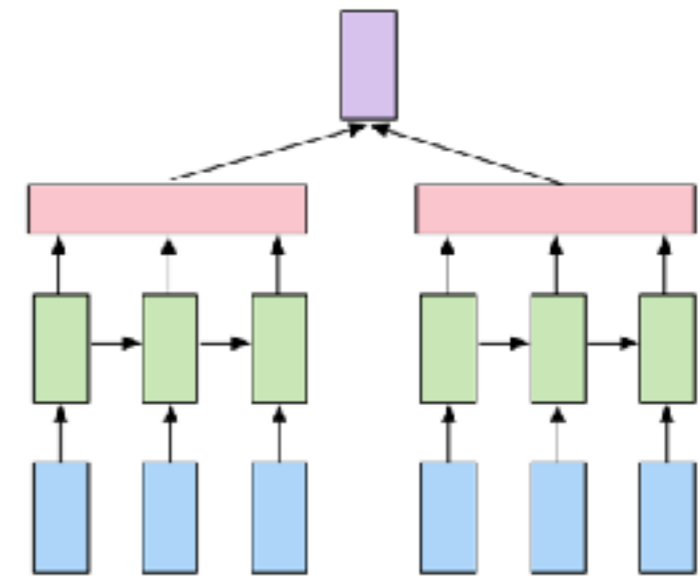
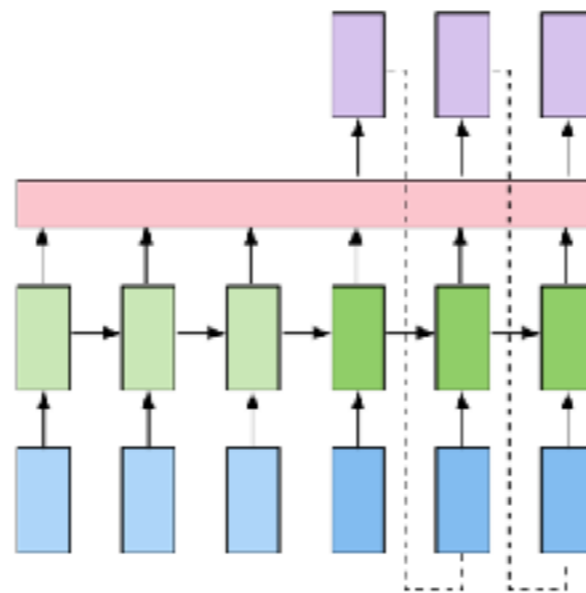
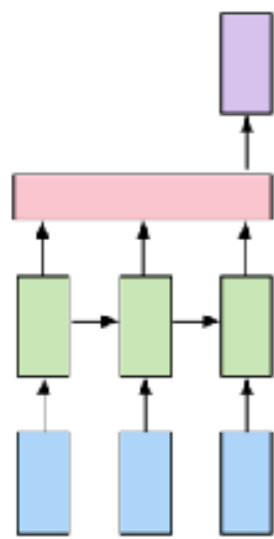
- **Dropout units at random:** is used as a regularization method to avoid overfitting
  - It allows to over-parameterize a network and still generalize well
- **Layer output normalization:** stabilizes the training process of a model, especially useful for self-attention architecture
- **Hyper-parameter optimization:** tuning well may have very huge impact in performance
- and more (e.g. initialization, label smoothing, weight decay)

# Putting everything together: Flexible modeling

- Sentiment classification
- Topic detection
- Spam detection
- Named Entity Recognition

- Machine translation
- Summarization
- Image captioning
- Conversational agents

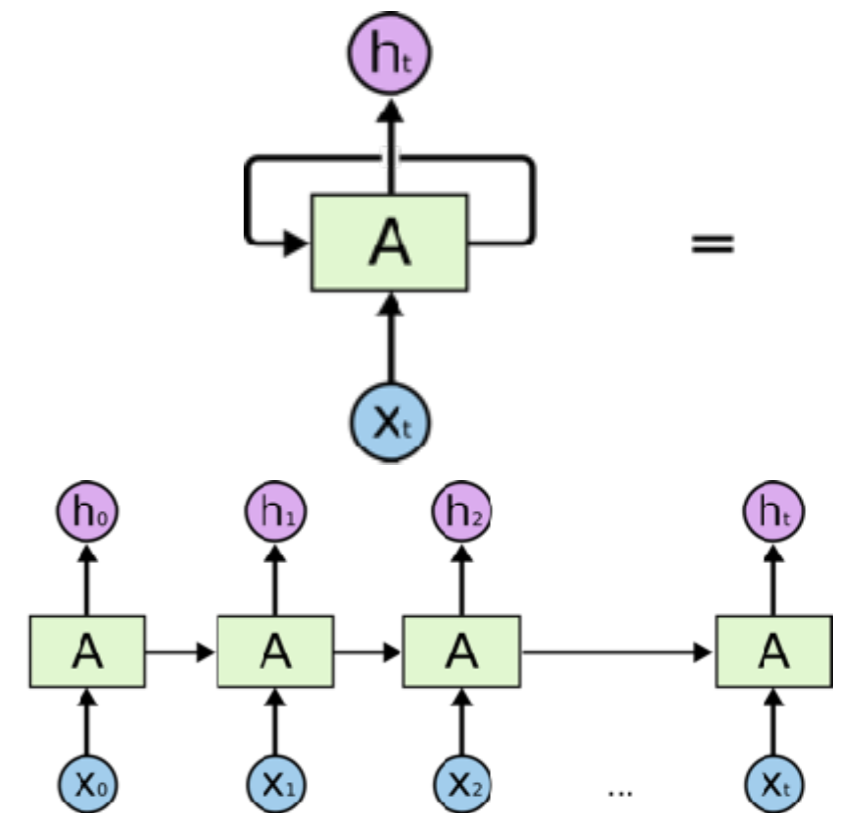
- Question answering
- Paraphrase detection
- Relation Extraction



- ✓ Multiple levels of abstraction (deep, hierarchical)
- ✓ End-to-end training with **stochastic gradient descent**
- ✓ Good basis for multi-task learning / transfer learning

# Outline of the talk

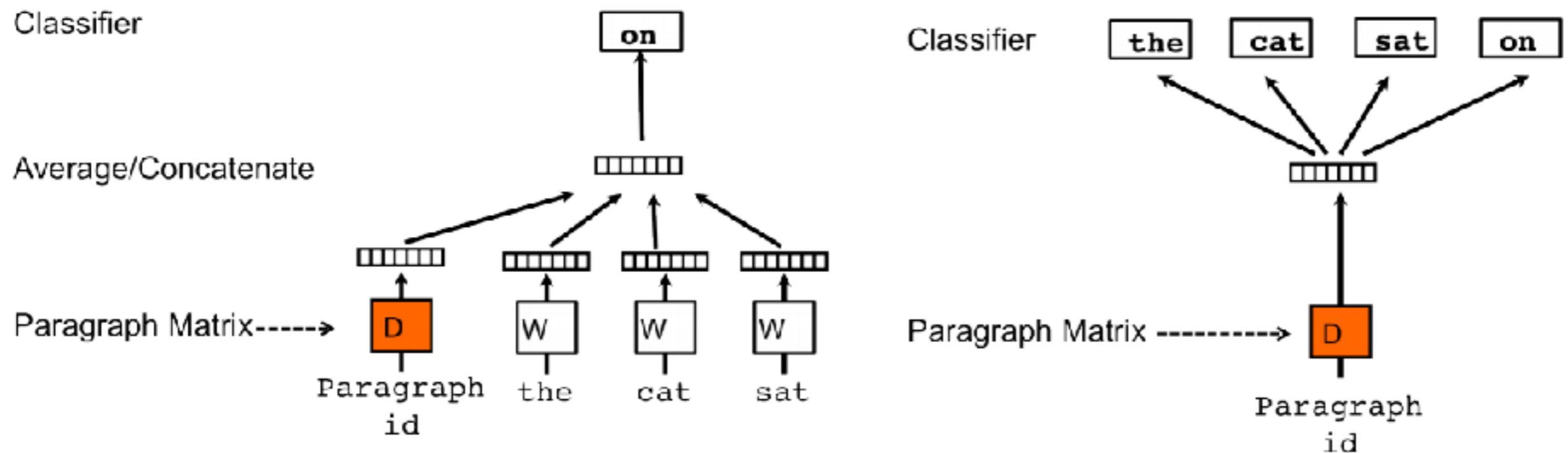
1. Introduction and Motivation
2. Word Representation Learning
  - Semantic similarity
  - Traditional and recent approaches
  - Intrinsic and extrinsic evaluation
3. Word Sequence Modeling
  - Essentials: RNNs, Attention, DL tricks
  - Text Classification
  - Machine Translation
4. Conclusion and Discussion



\* Figure from Colah's blog, 2015.

# Paragraph vectors for Document Classification

- Learning vectors of paragraphs inspired by word2vec
  - trained without supervision on a large corpus
  - preferably similar domain as the target
- **Two methods:** with or without word ordering



(Le et al., 2014)



# Paragraph vectors for Document Classification

- Learned paragraph vectors + logistic regression
- Outperformed previous method on sentence-level and document-level sentiment classification

*Table 1.* The performance of our method compared to other approaches on the Stanford Sentiment Treebank dataset. The error rates of other methods are reported in (Socher et al., 2013b).

Model	Error rate (Positive/Negative)	Error rate (Fine-grained)
Naïve Bayes (Socher et al., 2013b)	18.2 %	59.0%
SVMs (Socher et al., 2013b)	20.6%	59.3%
Bigram Naïve Bayes (Socher et al., 2013b)	16.9%	58.1%
Word Vector Averaging (Socher et al., 2013b)	19.9%	67.3%
Recursive Neural Network (Socher et al., 2013b)	17.6%	56.8%
Matrix Vector-RNN (Socher et al., 2013b)	17.1%	55.6%
Recursive Neural Tensor Network (Socher et al., 2013b)	14.6%	54.3%
<b>Paragraph Vector</b>	<b>12.2%</b>	<b>51.3%</b>

*Table 2.* The performance of Paragraph Vector compared to other approaches on the IMDB dataset. The error rates of other methods are reported in (Wang & Manning, 2012).

Model	Error rate
BoW (bnc) (Maas et al., 2011)	12.20 %
BoW (b $\Delta$ t'c) (Maas et al., 2011)	11.77%
LDA (Maas et al., 2011)	32.58%
Full+BoW (Maas et al., 2011)	11.67%
Full+Unlabeled+BoW (Maas et al., 2011)	11.11%
WRRBM (Dahl et al., 2012)	12.58%
WRRBM + BoW (bnc) (Dahl et al., 2012)	10.77%
MNB-uni (Wang & Manning, 2012)	16.45%
MNB-bi (Wang & Manning, 2012)	13.41%
SVM-uni (Wang & Manning, 2012)	13.05%
SVM-bi (Wang & Manning, 2012)	10.84%
NBSVM-uni (Wang & Manning, 2012)	11.71%
NBSVM-bi (Wang & Manning, 2012)	8.78%
<b>Paragraph Vector</b>	<b>7.42%</b>

(Le et al., 2014)

# Convolutional neural network for Document Classification

- Used multiple filter widths
- Dropout regularization (randomly dropping portion of hidden units during back-propagation)

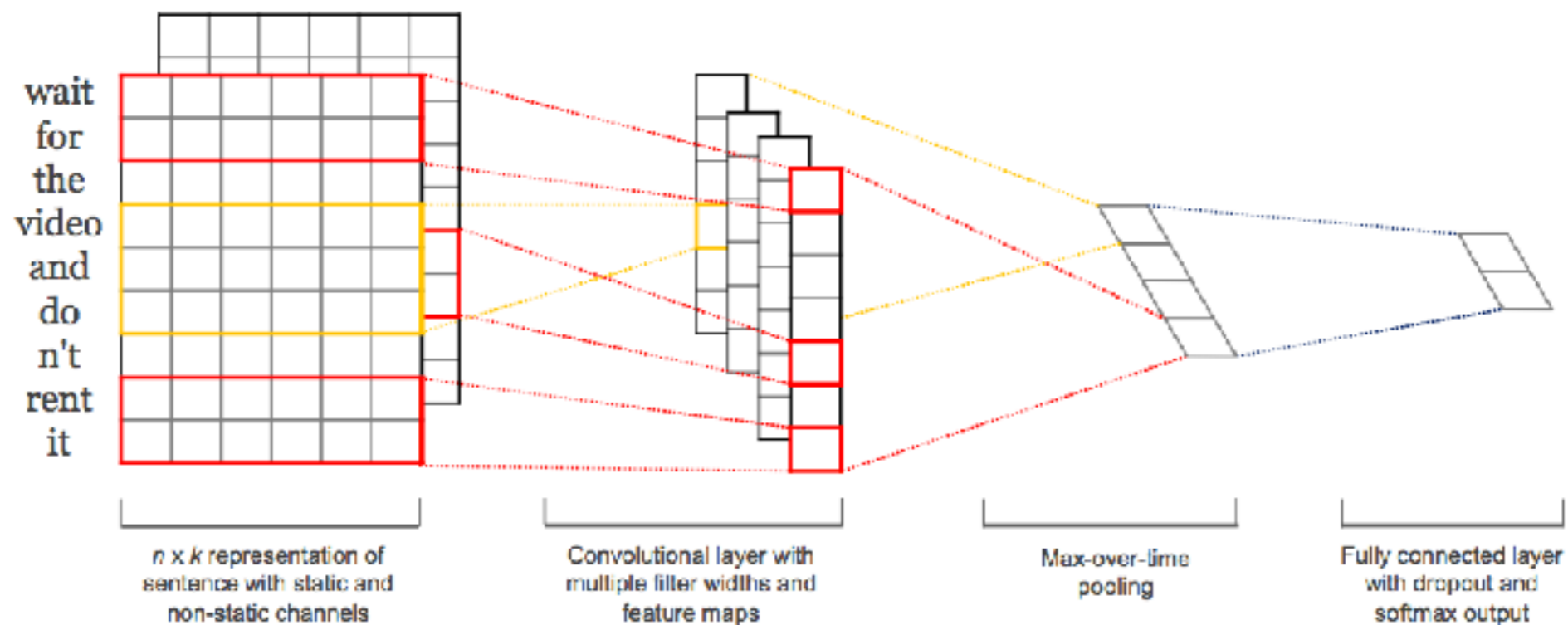


Figure 1: Model architecture with two channels for an example sentence.

(Kim et al., 2014)

# Convolutional neural network for Document Classification

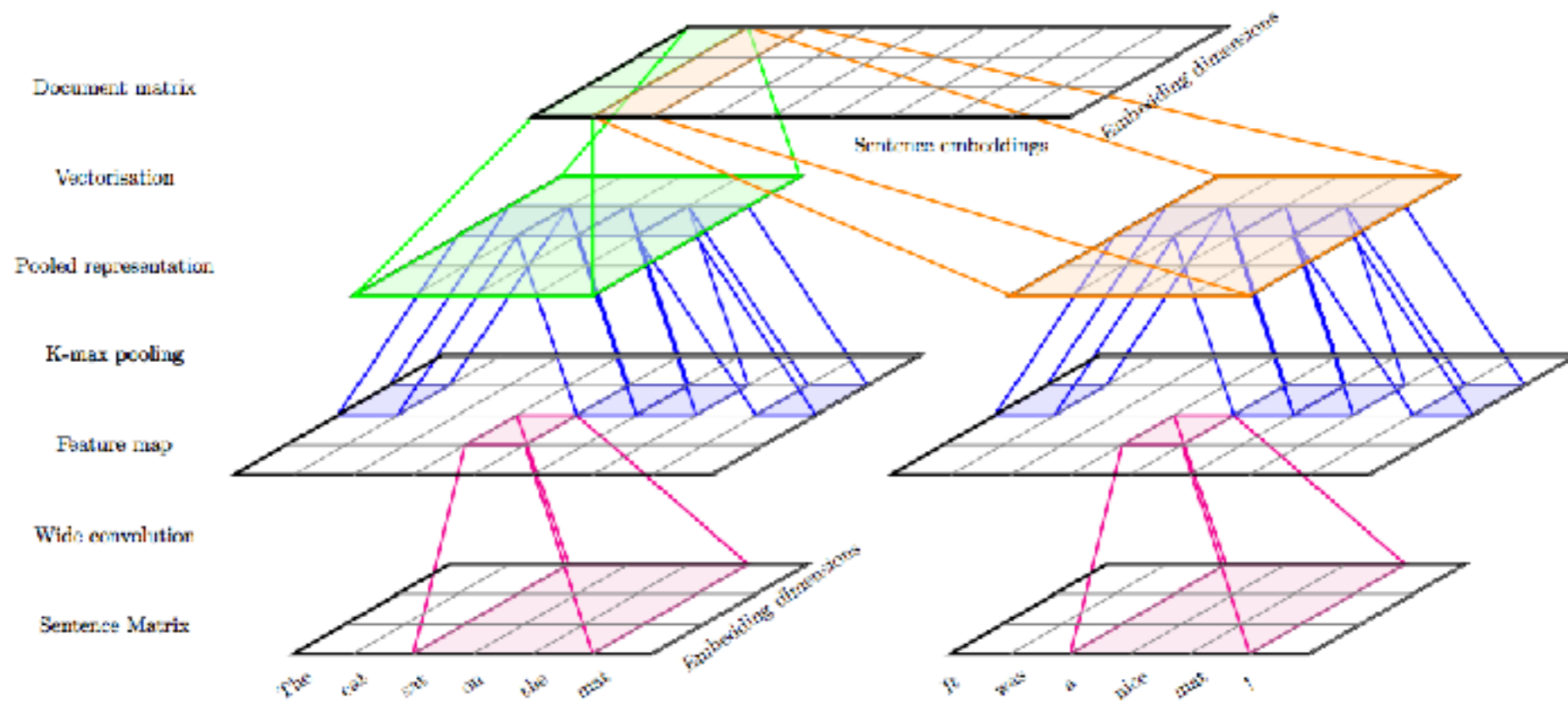
Model	MR	SST-1	SST-2	Subj	TREC	CR	MPQA
CNN-rand	76.1	45.0	82.7	89.6	91.2	79.8	83.4
CNN-static	81.0	45.5	86.8	93.0	92.8	84.7	<b>89.6</b>
CNN-non-static	<b>81.5</b>	48.0	87.2	93.4	93.6	84.3	89.5
CNN-multichannel	81.1	47.4	<b>88.1</b>	93.2	92.2	<b>85.0</b>	89.4
RAE (Socher et al., 2011)	77.7	43.2	82.4	—	—	—	86.4
MV-RNN (Socher et al., 2012)	79.0	44.4	82.9	—	—	—	—
RNTN (Socher et al., 2013)	—	45.7	85.4	—	—	—	—
DCNN (Kalchbrenner et al., 2014)	—	48.5	86.8	—	93.0	—	—
Paragraph-Vec (Le and Mikolov, 2014)	—	<b>48.7</b>	87.8	—	—	—	—
CCAE (Hermann and Blunsom, 2013)	77.8	—	—	—	—	—	87.2
Sent-Parser (Dong et al., 2014)	79.5	—	—	—	—	—	86.3
NBSVM (Wang and Manning, 2012)	79.4	—	—	93.2	—	81.8	86.3
MNB (Wang and Manning, 2012)	79.0	—	—	<b>93.6</b>	—	80.0	86.3
G-Dropout (Wang and Manning, 2013)	79.0	—	—	93.4	—	82.1	86.1
F-Dropout (Wang and Manning, 2013)	79.1	—	—	<b>93.6</b>	—	81.9	86.3
Tree-CRF (Nakagawa et al., 2010)	77.3	—	—	—	—	81.4	86.1
CRF-PR (Yang and Cardie, 2014)	—	—	—	—	—	82.7	—
SVM <sub>S</sub> (Silva et al., 2011)	—	—	—	—	<b>95.0</b>	—	—

- Not all baseline methods used drop-out though

(Kim et al., 2014)

# Modeling and Summarizing Documents with a Convolutional Network

- Similar to [Kim et al, 2014](#) however different
  - K-max pooling instead of max pooling
  - Two layers of convolutions



(Denil et al., 2014)



# Modeling and Summarizing Documents with a Convolutional Network

Model	Errors
SVM	66
BiNB	62
MaxEnt	61
Max-TDNN	76
NBoW	68
DCNN	45
Our Model	46

Model	Accuracy
BoW (b $\Delta$ t'c)	88.23%
Full+BoW	88.33%
Full+Unlabelled+BoW	88.89%
WRRBM	87.42%
WRRBM+BoW (bnc)	89.23%
SVM-bi	86.95%
NBSVM-uni	88.29%
NBSVM-bi	91.22%
Paragraph Vector	92.58%
Our model	89.38%

Table 1: **Left:** Number of test set errors on the twitter sentiment dataset. The first block of three entries is from Go *et al.* [5], the second block is from Kalchbrenner *et al.* [13]. **Right:** Error rates on the IMDB movie review data set. The first block is from Maas *et al.* [16], the second from Dahl *et al.* [3], the third from Wang and Manning [24] and the fourth from Le and Mikolov [15].

(Denil et al., 2014)

# Modeling and Summarizing Documents with a Convolutional Network

Proportion	Summary	Random	Margin	Fixed	Summary	Random	Margin
100%	83.03	83.03	—				
50%	83.53	79.79	+3.74	Pick 5	83.07	80.02	+3.05
33%	83.10	76.72	+6.38	Pick 4	83.09	79.05	+4.04
25%	82.91	74.87	+8.04	Pick 3	82.88	77.15	+5.73
20%	82.67	73.20	+9.47	Pick 2	82.04	74.48	+7.56
First and last	68.62						

Table 2: Results of classifying summaries with Naïve Bayes. Results labelled proportion indicate selecting up to the indicated percentage of sentences in the review, and results labelled fixed show the result of selecting a fixed number of sentences from each. The summary column shows the accuracy of Naïve Bayes on summaries produced by our model. The random column shows the same model classifying summaries created by selecting sentences at random. The margin column shows the difference in accuracy between our model and the random summaries.

(Denil et al., 2014)



# Modeling and Summarizing Documents with a Convolutional Network

Graphics is far from the best part of the game. **This is the number one best TH game in the series.** Next to Underground. **It deserves strong love. It is an insane game.** There are massive levels, massive unlockable characters... it's just a massive game. **Waste your money on this game. This is the kind of money that is wasted properly.** And even though graphics suck, that doesn't make a game good. Actually, the graphics were good at the time. Today the graphics are crap. WHO CARES? As they say in Canada, This is the fun game, aye. (You get to go to Canada in THPS3) Well, I don't know if they say that, but they might. who knows. Well, Canadian people do. Wait a minute, I'm getting off topic. This game rocks. Buy it, play it, enjoy it, love it. It's PURE BRILLIANCE.

The first was good and original. I was a not bad horror/comedy movie. So I heard a second one was made and I had to watch it . What really makes this movie work is Judd Nelson's character and the sometimes clever script. **A pretty good script for a person who wrote the Final Destination films and the direction was okay.** Sometimes there's scenes where it looks like it was filmed using a home video camera with a grainy - look. Great made - for - TV movie. **It was worth the rental and probably worth buying just to get that nice eerie feeling and watch Judd Nelson's Stanley doing what he does best.** I suggest newcomers to watch the first one before watching the sequel, just so you'll have an idea what Stanley is like and get a little history background.

When the movie was released it was the biggest hit and it soon became the Blockbuster. But honestly the movie is a ridiculous watch with a plot which glorifies a loser. The movie has a Tag - line - "Preeti Madhura, Tyaga Amara" which means Love's Sweet but Sacrifice is Immortal. **In the movie the hero of the movie (Ganesh) sacrifices his love for the leading lady (Pooja Gandhi) even though the two loved each other!** His justification is the meaning of the tag - line. This movie influenced so many young broken hearts that they found this "Loser - like Sacrificial" attitude very thoughtful and hence became the cult movie it is, when they could have moved on with their lives. **Ganesh's acting in the movie is Amateurish, Crass and Childishly stupid.** He actually looks funny in a song, (Onde Ondu Sari ...) when he's supposed to look all stylish and cool. His looks don't help the leading role either. **His hair style is badly done in most part of the movie. POOJA GANDHI CANT ACT.** Her costumes are horrendous in the movie and very inconsistent. **The good part about the movie is the excellent cinematography and brilliant music by Mano Murthy which are actually the true saving graces of the movie.** Also the lyrics by Jayant Kaikini are very well penned. The Director Yograj Bhat has to be lauded picturization the songs in a tasteful manner. Anyway all - in - all except for the songs, the movie is a very ordinary one !!!!!

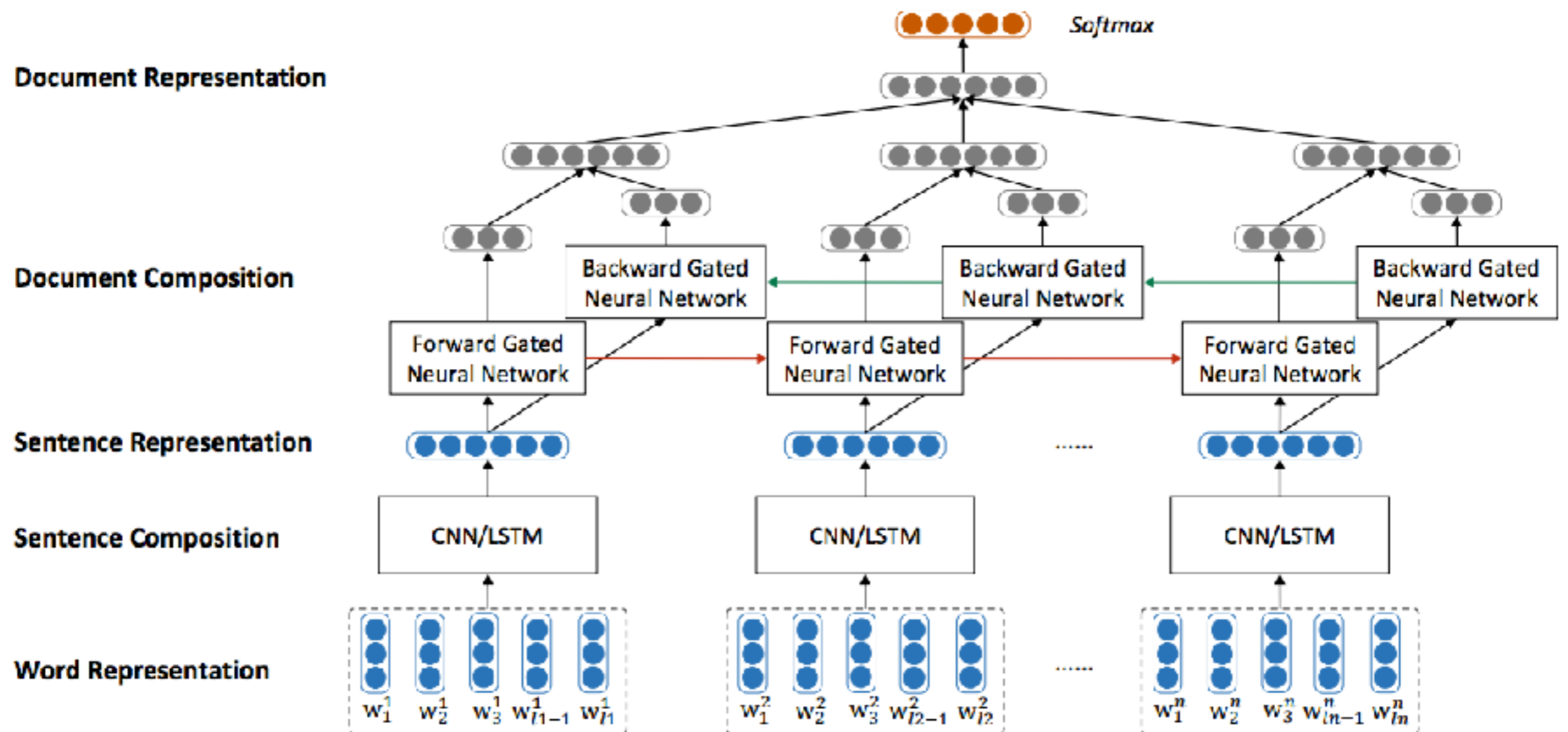
A friend and I went through a phase some (alot of) years ago of selecting the crappiest horror films in the video shop for an evening's entertainment. For some reason, I ended up buying this one (probably v. v. cheap). **The cheap synth soundtrack is a classic of its time and genre.** There's also a few very amusing scenes. Among them is a scene where a man's being attacked and defends himself with a number of unlikely objects, it made me laugh at the time (doesn't seem quite so funny in retrospect but there you go). **Apart from that it's total crap, mind you.** But probably worth a watch if you like films like "Chopping Mall". Yes, I've seen that too.

I tried restarting the movie twice. I put it in three machines to see what was wrong . Did Steven Seagal's voice change? **Did he die during filming and the studio have to dub the sound with someone who doesn't even resemble him?** Or was the sound on the DVD destroyed? After about 10 minutes, you finally hear the actor's real voice. Though throughout most of the film, it sounds like the audio was recorded in a bathroom. **I would be ashamed to donate a copy of this movie to Goodwill, if I owned a copy.** I rented it, but I will never do that again. I will check this database before renting any more of his movies, all of which were (more or less) good movies. **You usually knew what you were getting when you watched a Steven Seagal movie.** I guess that is no more.

Vertigo co - stars Stewart (in his last turn as a romantic lead) and Novak elevate this, Stewart's other "Christmas movie," movie to above mid - level entertainment. **The chemistry between the two stars makes for a fairly moving experience and further revelation can be gleaned from the movie if witchcraft is seen as a metaphor for the private pain that hampers many people's relationships.** All in all, a nice diversion with legendary stars, 7/10

(Denil et al., 2014)

# Gated recurrent neural network for Document Classification



(Tang et al., 2015)



# Gated recurrent neural network for Document Classification

	Yelp 2013		Yelp 2014		Yelp 2015		IMDB	
	Accuracy	MSE	Accuracy	MSE	Accuracy	MSE	Accuracy	MSE
Majority	0.356	3.06	0.361	3.28	0.369	3.30	0.179	17.46
SVM + Unigrams	0.589	0.79	0.600	0.78	0.611	0.75	0.399	4.23
SVM + Bigrams	0.576	0.75	0.616	0.65	0.624	0.63	0.409	3.74
SVM + TextFeatures	0.598	0.68	0.618	0.63	0.624	0.60	0.405	3.56
SVM + AverageSG	0.543	1.11	0.557	1.08	0.568	1.04	0.319	5.57
SVM + SSWE	0.535	1.12	0.543	1.13	0.554	1.11	0.262	9.16
JMARS	N/A	–	N/A	–	N/A	–	N/A	4.97
Paragraph Vector	0.577	0.86	0.592	0.70	0.605	0.61	0.341	4.69
Convolutional NN	0.597	0.76	0.610	0.68	0.615	0.68	0.376	3.30
Conv-GRNN	0.637	0.56	0.655	0.51	0.660	0.50	0.425	<b>2.71</b>
LSTM-GRNN	<b>0.651</b>	<b>0.50</b>	<b>0.671</b>	<b>0.48</b>	<b>0.676</b>	<b>0.49</b>	<b>0.453</b>	3.00

Table 2: Sentiment classification on Yelp 2013/2014/2015 and IMDB datasets. Evaluation metrics are accuracy (higher is better) and MSE (lower is better). The best method in each setting is in **bold**.

(Tang et al., 2015)

# Hierarchical attention networks for Document Classification

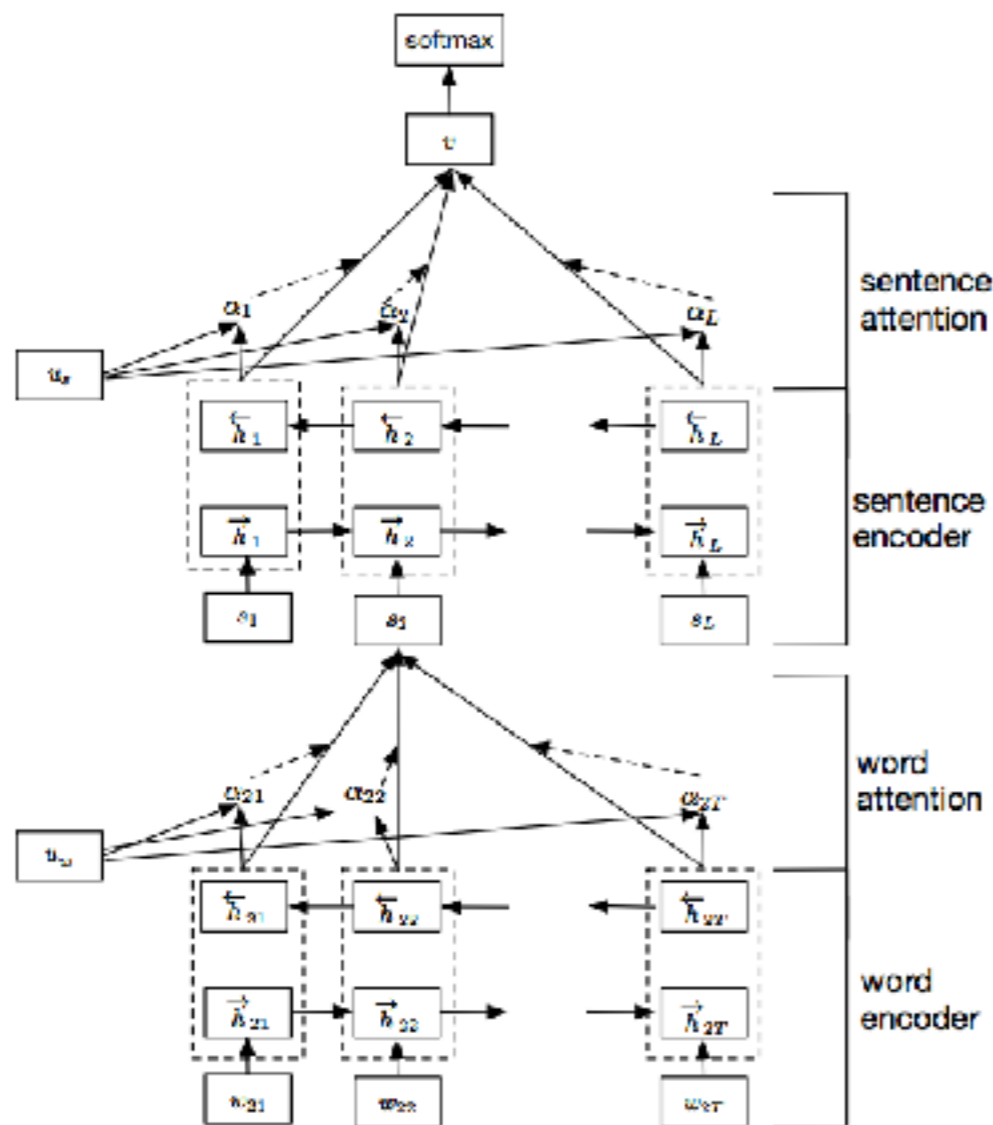


Figure 2: Hierarchical Attention Network.

- Very similar hierarchical structure as [Tang et al., 2015](#) except average pooling
- attention mechanism at the word and document levels

$$u_{it} = \tanh(W_w h_{it} + b_w)$$

$$\alpha_{it} = \frac{\exp(u_{it}^\top u_w)}{\sum_t \exp(u_{it}^\top u_w)}$$

$$s_i = \sum_t \alpha_{it} h_{it}.$$

(Yang et al., 2016)

# Hierarchical attention networks for Document Classification

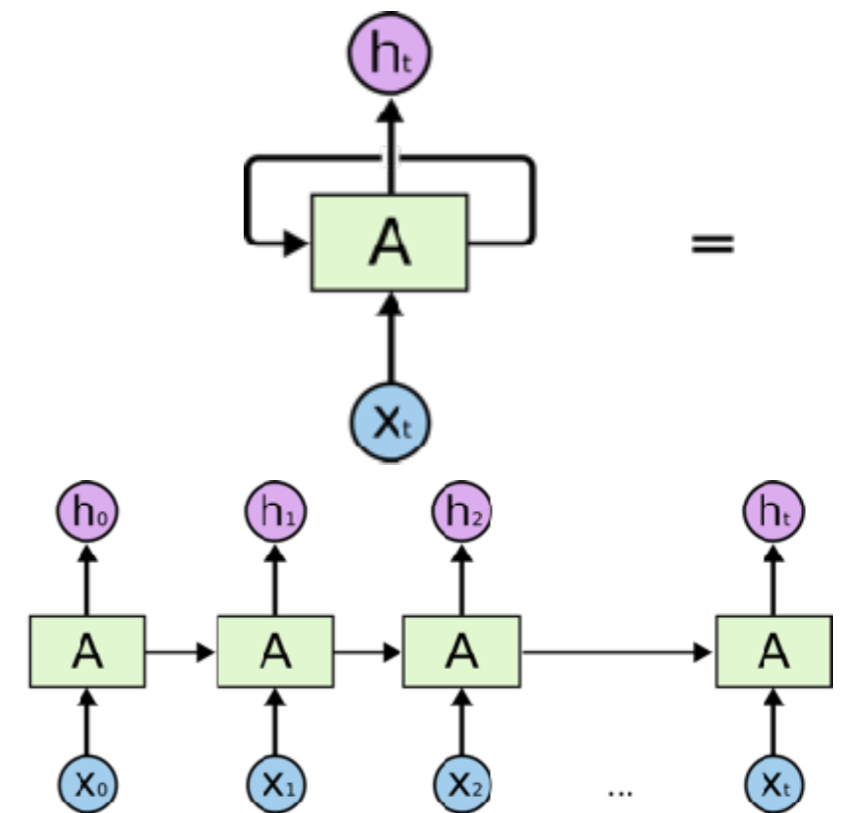
	Methods	Yelp'13	Yelp'14	Yelp'15	IMDB	Yahoo Answer	Amazon
<b>Zhang et al., 2015</b>	BoW	-	-	58.0	-	68.9	54.4
	BoW TFIDF	-	-	59.9	-	71.0	55.3
	ngrams	-	-	56.3	-	68.5	54.3
	ngrams TFIDF	-	-	54.8	-	68.5	52.4
	Bag-of-means	-	-	52.5	-	60.5	44.1
<b>Tang et al., 2015</b>	Majority	35.6	36.1	36.9	17.9	-	-
	SVM + Unigrams	58.9	60.0	61.1	39.9	-	-
	SVM + Bigrams	57.6	61.6	62.4	40.9	-	-
	SVM + TextFeatures	59.8	61.8	62.4	40.5	-	-
	SVM + AverageSG	54.3	55.7	56.8	31.9	-	-
	SVM + SSWE	53.5	54.3	55.4	26.2	-	-
<b>Zhang et al., 2015</b>	LSTM	-	-	58.2	-	70.8	59.4
	CNN-char	-	-	62.0	-	71.2	59.6
	CNN-word	-	-	60.5	-	71.2	57.6
<b>Tang et al., 2015</b>	Paragraph Vector	57.7	59.2	60.5	34.1	-	-
	CNN-word	59.7	61.0	61.5	37.6	-	-
	Conv-GRNN	63.7	65.5	66.0	42.5	-	-
	LSTM-GRNN	65.1	67.1	67.6	45.3	-	-
<b>This paper</b>	HN-AVE	67.0	69.3	69.9	47.8	75.2	62.9
	HN-MAX	66.9	69.3	70.1	48.2	75.2	62.9
	HN-ATT	<b>68.2</b>	<b>70.5</b>	<b>71.0</b>	<b>49.4</b>	<b>75.8</b>	<b>63.6</b>

**Table 2:** Document Classification, in percentage

(Yang et al., 2016)

# Outline of the talk

1. Introduction and Motivation
2. Word Representation Learning
  - Semantic similarity
  - Traditional and recent approaches
  - Intrinsic and extrinsic evaluation
3. Word Sequence Modeling
  - Essentials: RNNs, Attention, DL tricks
  - Text Classification
  - Machine Translation
4. Conclusion and Discussion



\* Figure from Colah's blog, 2015.



# RNN encoder-decoder for Machine Translation

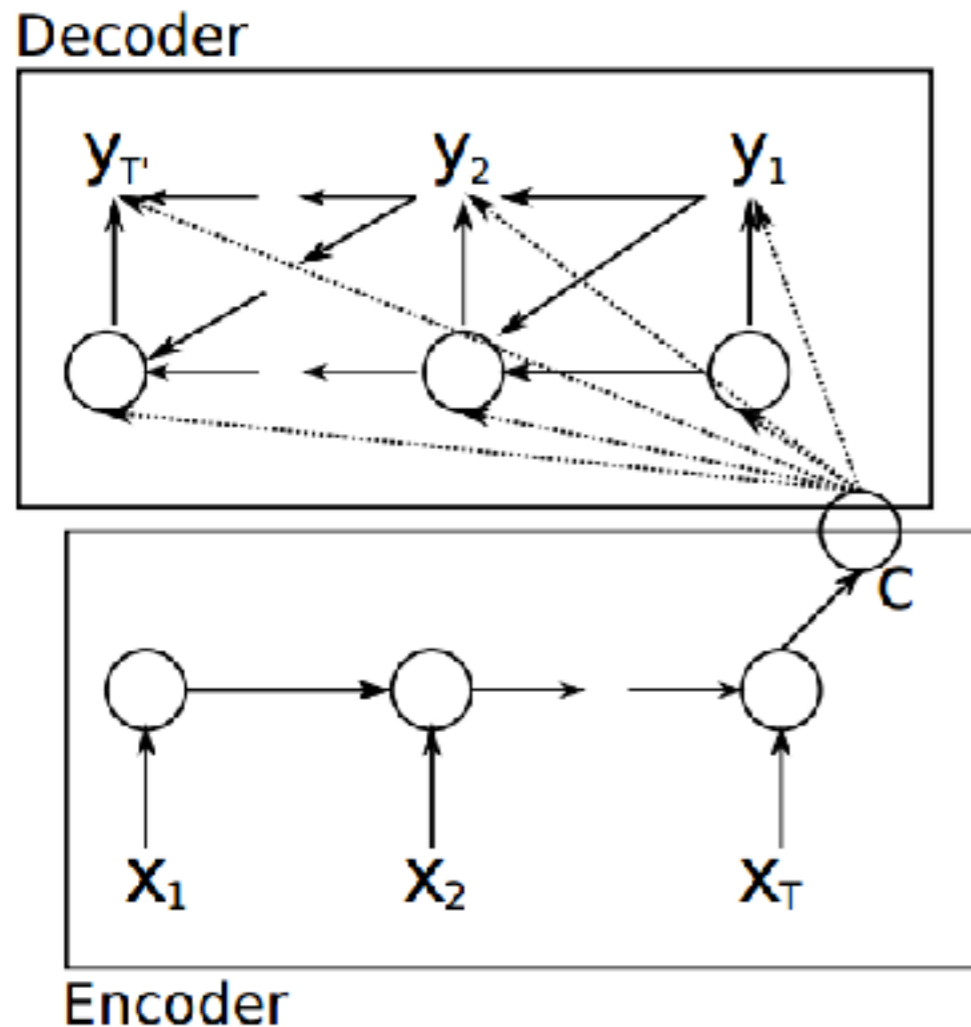


Figure 1: An illustration of the proposed RNN Encoder-Decoder.

- GRU as hidden layer
- Maximize the log likelihood of the target sequence given the source sequence:

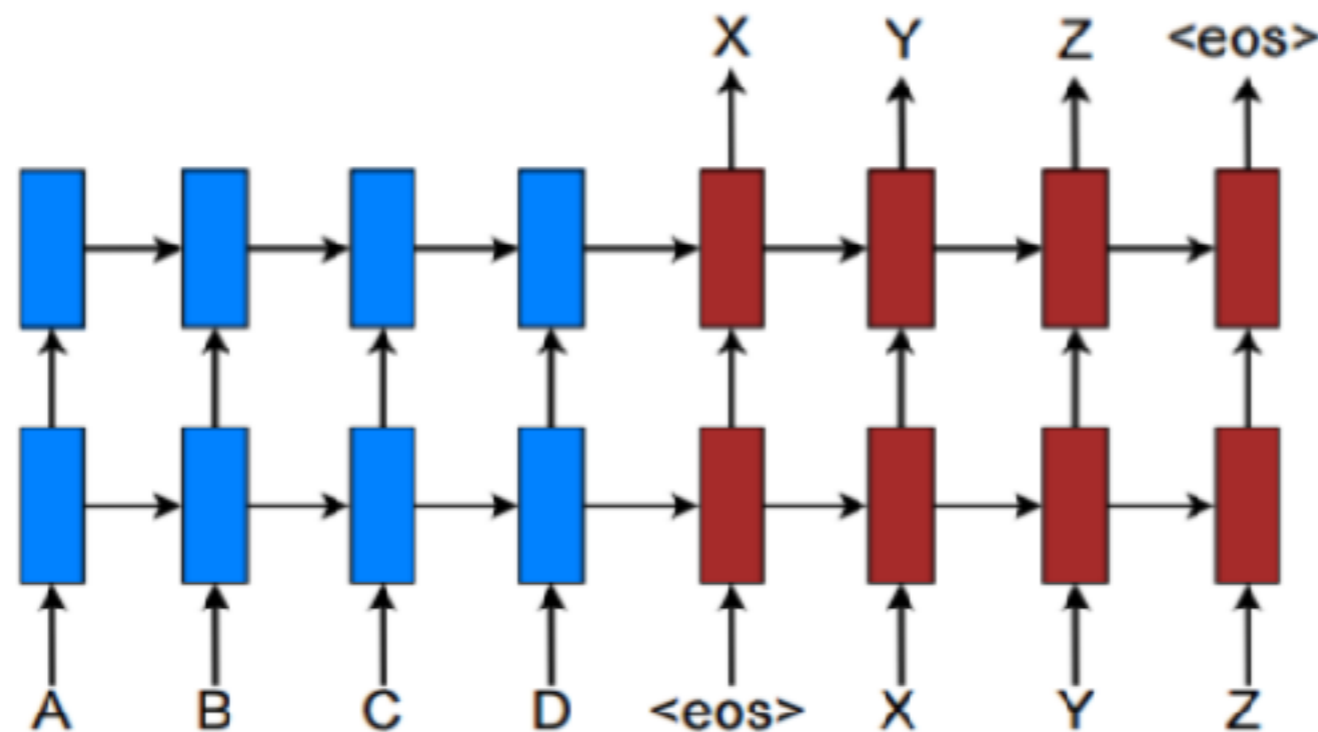
$$\max_{\theta} \frac{1}{N} \sum_{n=1}^N \log p_{\theta}(Y_n | X_n)$$

- WMT 2014 (EN→FR)

Models	BLEU	
	dev	test
Baseline	30.64	33.30
RNN	31.20	33.87
CSLM + RNN	31.48	34.64
CSLM + RNN + WP	31.50	34.54

(Cho et al., 2014)

# Sequence to sequence learning for Machine Translation



- LSTM hidden layers instead of GRU
- 4 layers deep instead of shallow encoder-decoder

(Sutskever et al., 2014)

# Sequence to sequence learning for Machine Translation

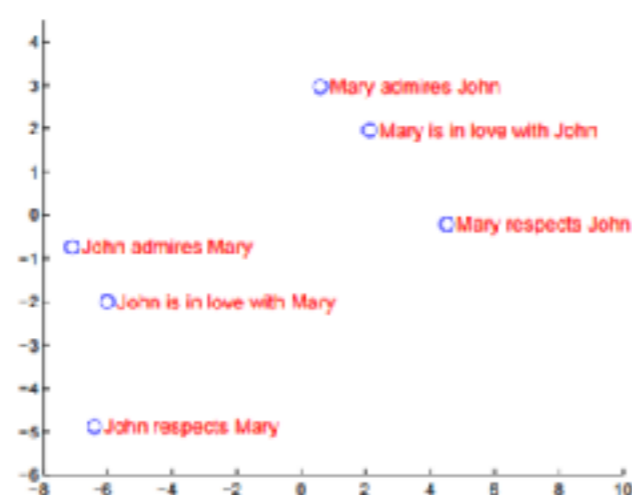
- WMT 2014 (EN→FR)

Trick-1: Reverse  
the input  
sequence.

Method	test BLEU score (ntst14)
Bahdanau et al. [2]	28.45
Baseline System [29]	33.30
Single forward LSTM, beam size 12	26.17
Single reversed LSTM, beam size 12	30.59
Ensemble of 5 reversed LSTMs, beam size 1	33.00
Ensemble of 2 reversed LSTMs, beam size 12	33.27
Ensemble of 5 reversed LSTMs, beam size 2	34.50
Ensemble of 5 reversed LSTMs, beam size 12	34.81

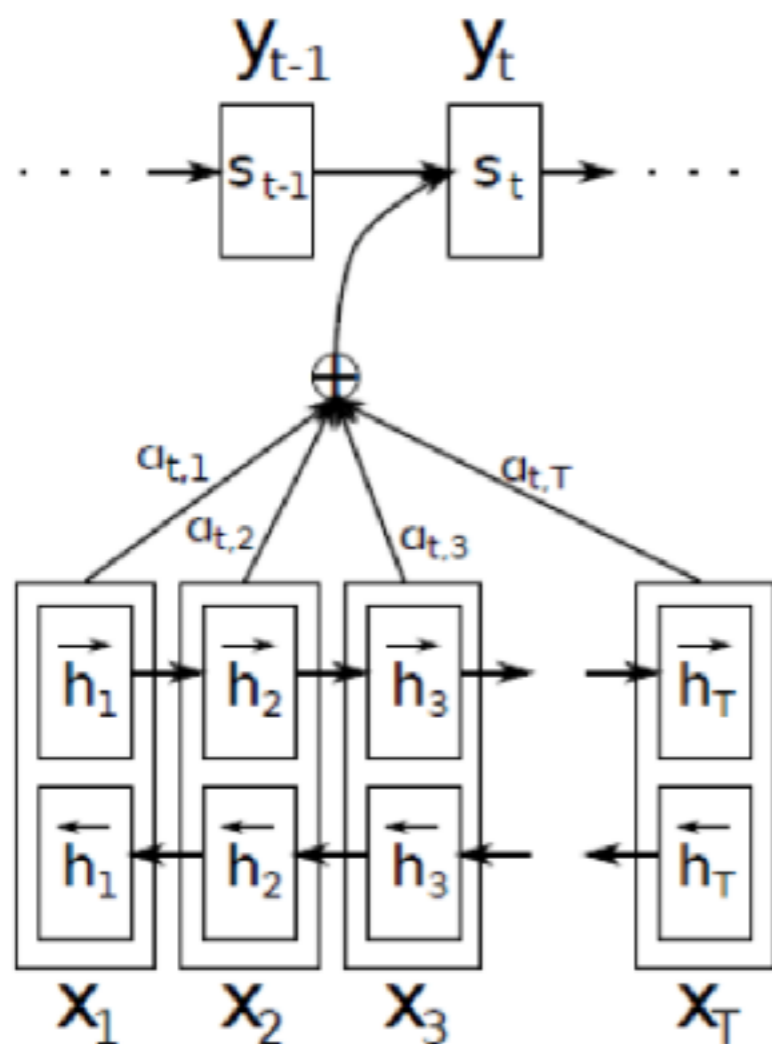
Trick-2: Ensemble  
Neural Nets.

- PCA projection of the hidden state of the last encoder layer



(Sutskever et al., 2014)

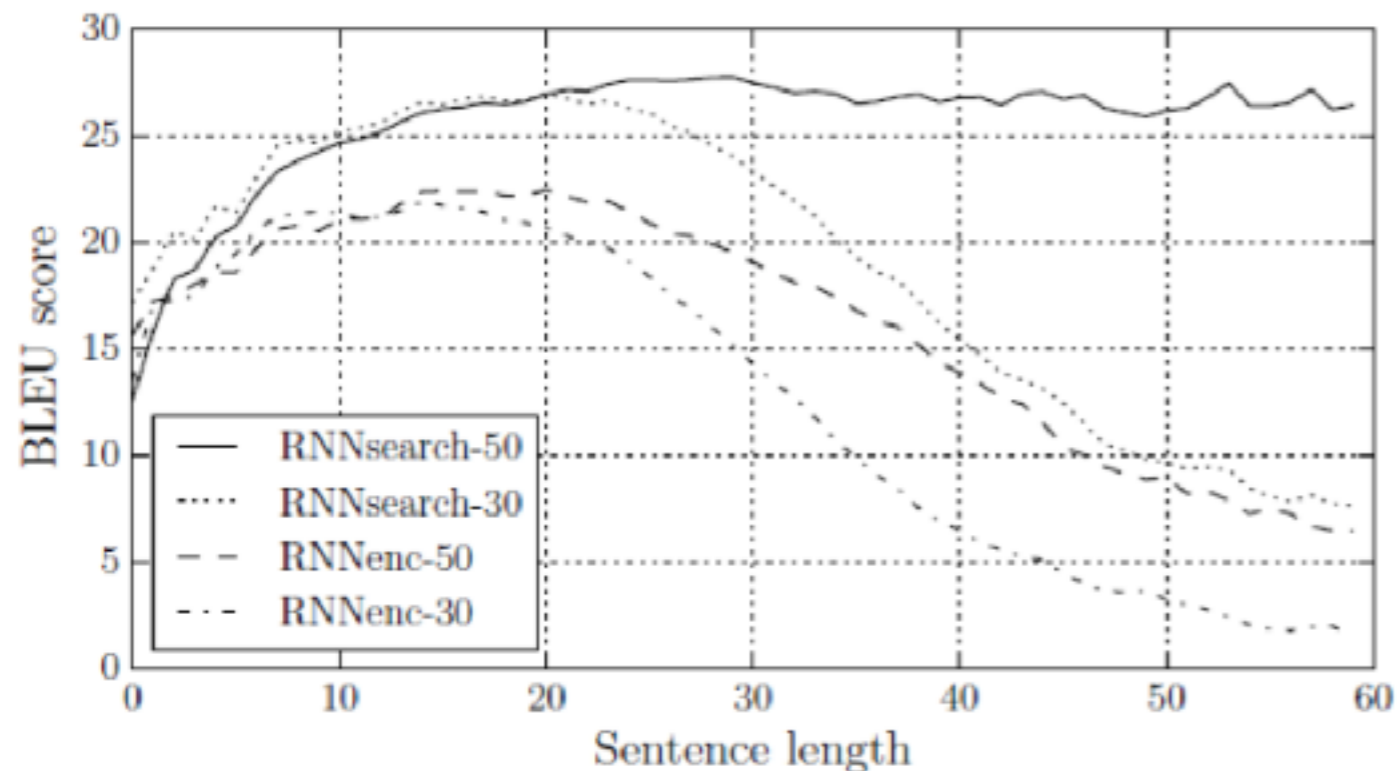
# Jointly learning to align and translate for Machine Translation



- **Limitation:** can we compress all the needed information in the last encoder state?
- **Idea:** use all the hidden states of the encoder
  - length proportional to that of the sentence!
  - compute a weighted average of all the hidden states

(Bahdanau et al., 2015)

# Jointly learning to align and translate for Machine Translation



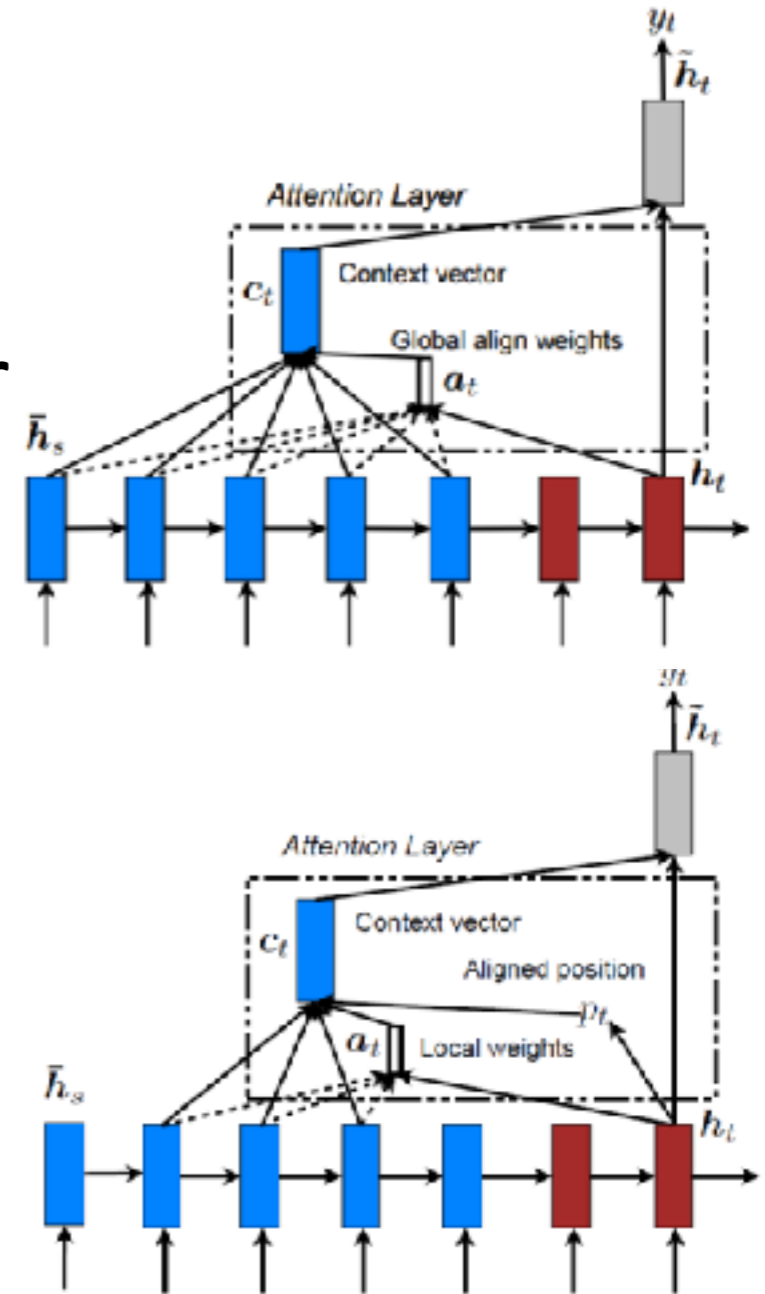
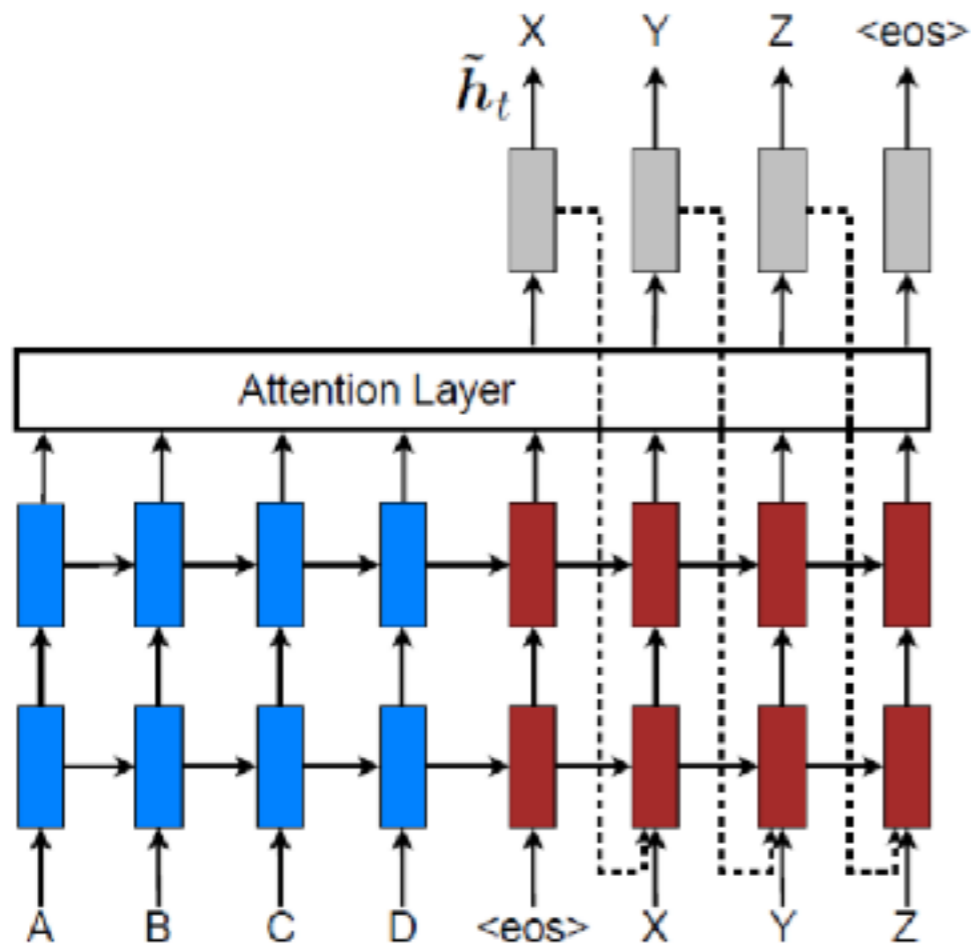
- WMT 2014 (EN→FR)

Model	All	No UNK <sup>o</sup>
RNNencdec-30	13.93	24.19
RNNsearch-30	21.50	31.44
RNNencdec-50	17.82	26.71
RNNsearch-50	26.75	34.16
RNNsearch-50*	28.45	36.15
Moses	33.30	35.63

(Bahdanau et al., 2015)

# Effective approaches to attention-based NMT

- Global and local attention
- Input-feeding approach
- Stacked LSTM instead of single-layer

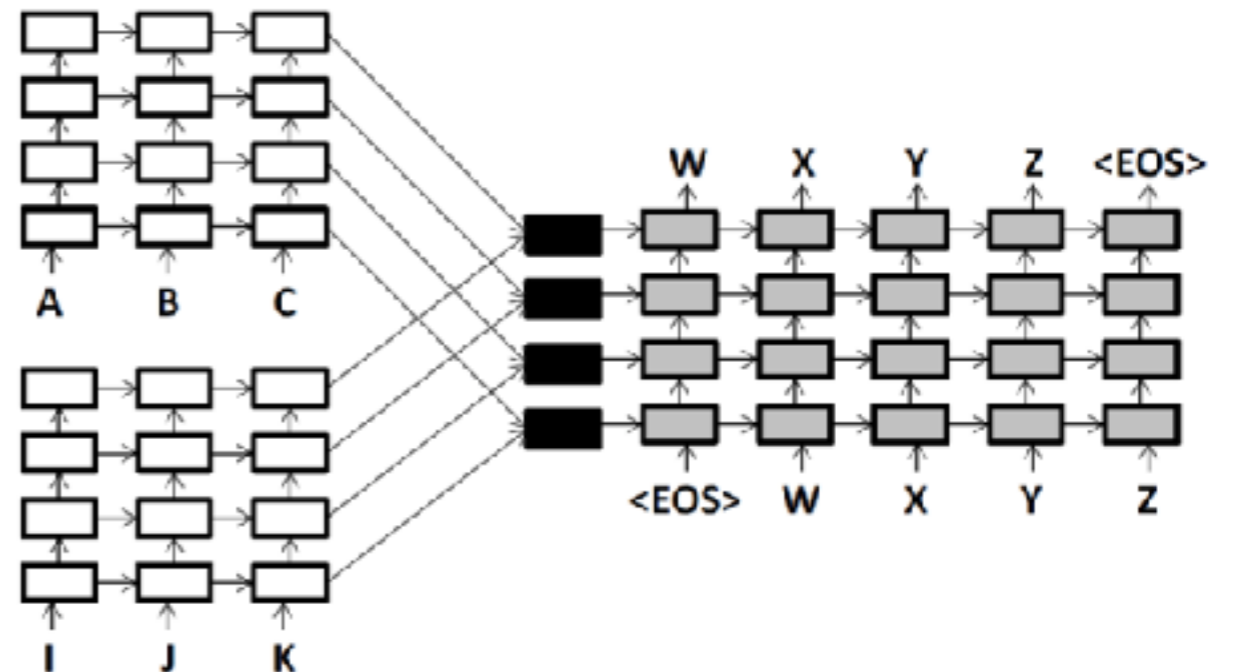


(Luong et al., 2015)



# Multi-source NMT

- Train  $p(e|f, g)$  model directly on trilingual data
- Use it to decode  $e$  given any  $(f, g)$  pair
- Take local-attention NMT model and concatenate context from multiple sources



Source 1: UNK Aspekte sind ebenfalls wichtig .

Target: UNK aspects are important , too .

Source 2: Les aspects UNK sont également importants .

(Zoph and Knight, 2016)

# Multi-source NMT

- Multi-source training improves over individual French English and German English pairs
  - **Best:** basic concatenation with attention

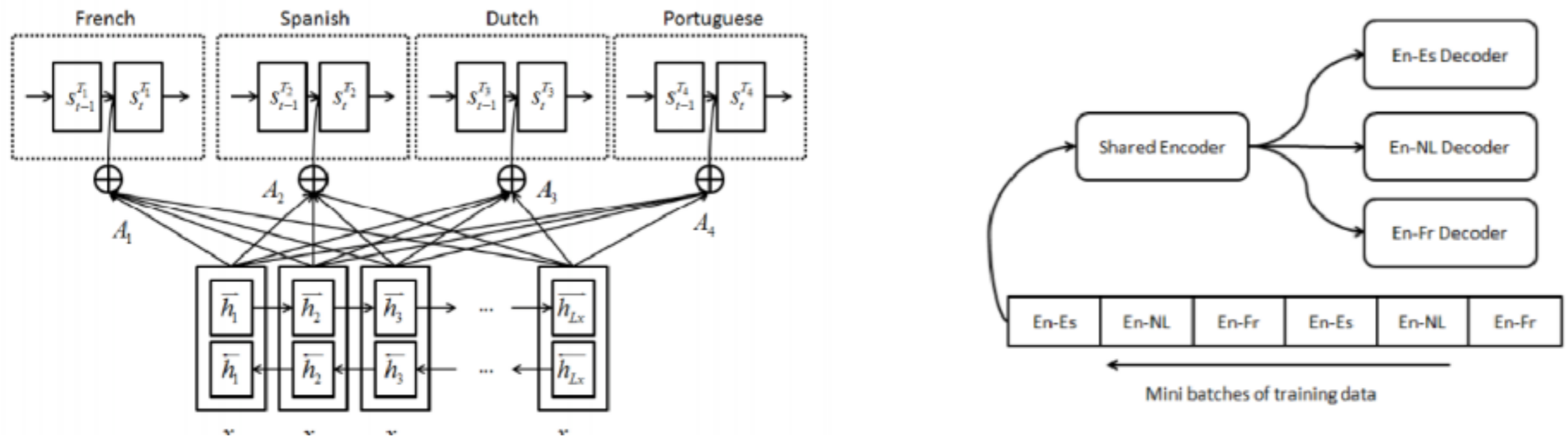
Target = English			
Source	Method	Ppl	BLEU
French	—	10.3	21.0
German	—	15.9	17.3
French+German	Basic	8.7	23.2
French+German	Child-Sum	9.0	22.5
French+French	Child-Sum	10.9	20.7
French	Attention	8.1	25.2
French+German	B-Attent.	5.7	30.0
French+German	CS-Attent.	6.0	29.6

Target = German			
Source	Method	Ppl	BLEU
French	—	12.3	10.6
English	—	9.6	13.4
French+English	Basic	9.1	14.5
French+English	Child-Sum	9.5	14.4
English	Attention	7.3	17.6
French+English	B-Attent.	6.9	18.6
French+English	CS-Attent.	7.1	18.2

(Zoph and Knight, 2016)

# Multi-target NMT

- Multi-task learning framework for multiple target language translation
  - Optimization for one to many model

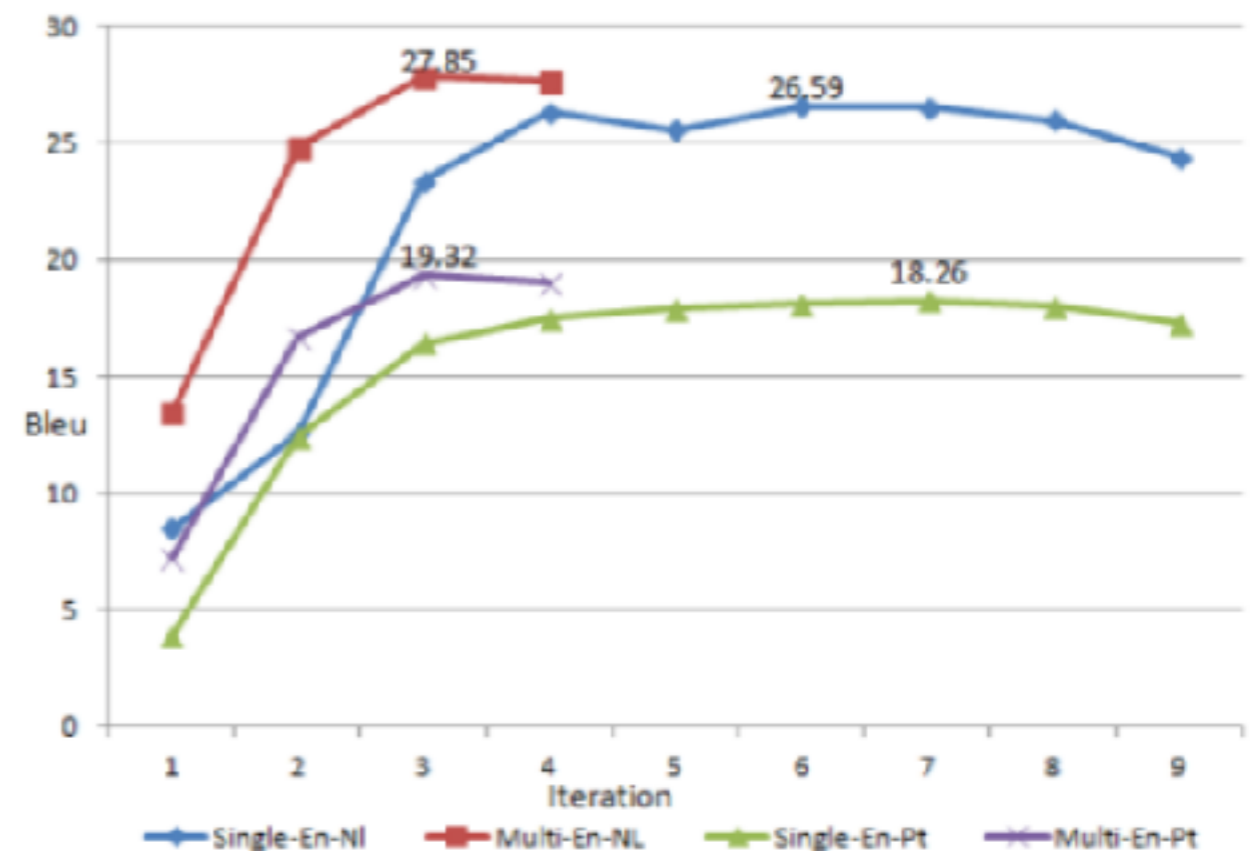


(Dong et al., 2015)

# Multi-target NMT

- Improves over NMT and Moses baselines over WMT 2013 test
  - but also on larger datasets
- Faster and better convergence in multiple language translation

	Nmt Baseline	Nmt Multi-Full	Nmt Multi-Partial	Moses
En-Fr	23.89	26.02(+2.13)	25.01(+1.12)	23.83
En-Es	23.28	25.31(+2.03)	25.83(+2.55)	23.58



(Dong et al., 2015)

# Multi-way, Multilingual NMT

- Encoder-decoder model with multiple encoders and decoders shared across language pairs
  - share knowledge through a universal space
  - good for low-resource langs
- Attention is pair specific, **hence expensive**  $O(L^2)$ 
  - instead share attention across all pairs!

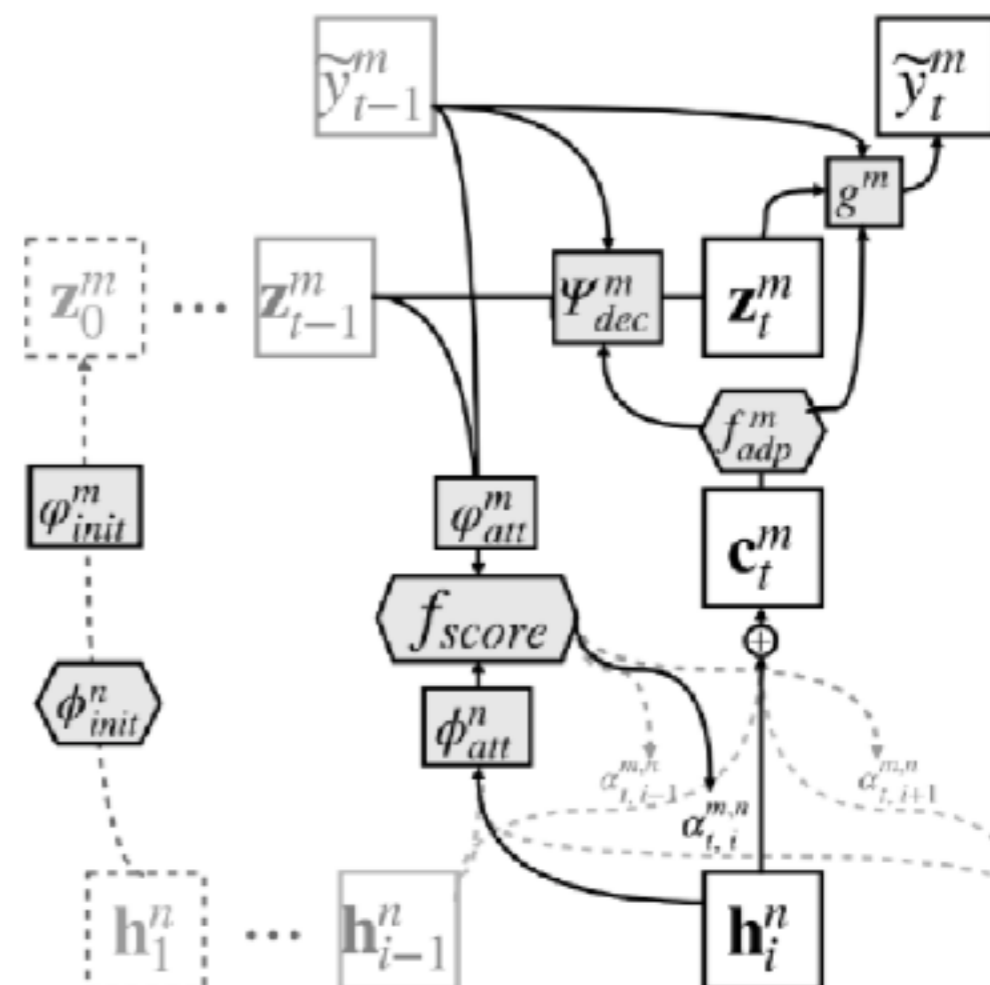


Figure:  $n$ \_th encoder and  $m$ \_th decoder at timestep  $t$  /  $\phi$  makes encoder & decoder states compatible with the attention mechanism /  $f_{adp}$  makes context vector compatible with the decoder

→ all these transformations to support different types of encoders/decoders for different languages!

(Firat et al., 2016)



# Multi-way, Multilingual NMT

	Size	Single	Single+DF	Multi
En→Fi	100k	5.06/3.96	4.98/3.99	6.2/ <b>5.17</b>
	200k	7.1/6.16	7.21/6.17	8.84/ <b>7.53</b>
	400k	9.11/7.85	9.31/8.18	11.09/ <b>9.98</b>
	800k	11.08/9.96	11.59/10.15	12.73/ <b>11.28</b>
De→En	210k	14.27/13.2	14.65/13.88	16.96/ <b>16.26</b>
	420k	18.32/17.32	18.51/17.62	19.81/ <b>19.63</b>
	840k	21/19.93	21.69/20.75	22.17/ <b>21.93</b>
	1.68m	23.38/23.01	23.33/22.86	23.86/ <b>23.52</b>
En→De	210k	11.44/11.57	11.71/11.16	12.63/ <b>12.68</b>
	420k	14.28/14.25	14.88/15.05	15.01/ <b>15.67</b>
	840k	17.09/17.44	17.21/17.88	17.33/ <b>18.14</b>
	1.68m	19.09/19.6	19.36/20.13	19.23/ <b>20.59</b>

- **Consistent improvements for low-resource languages**
  - the lower the training data the bigger the improvement
- **In large-scale translation improves only translation to English**
  - hypothesis: EN appears always as source or target language for all pairs → better decoder ?

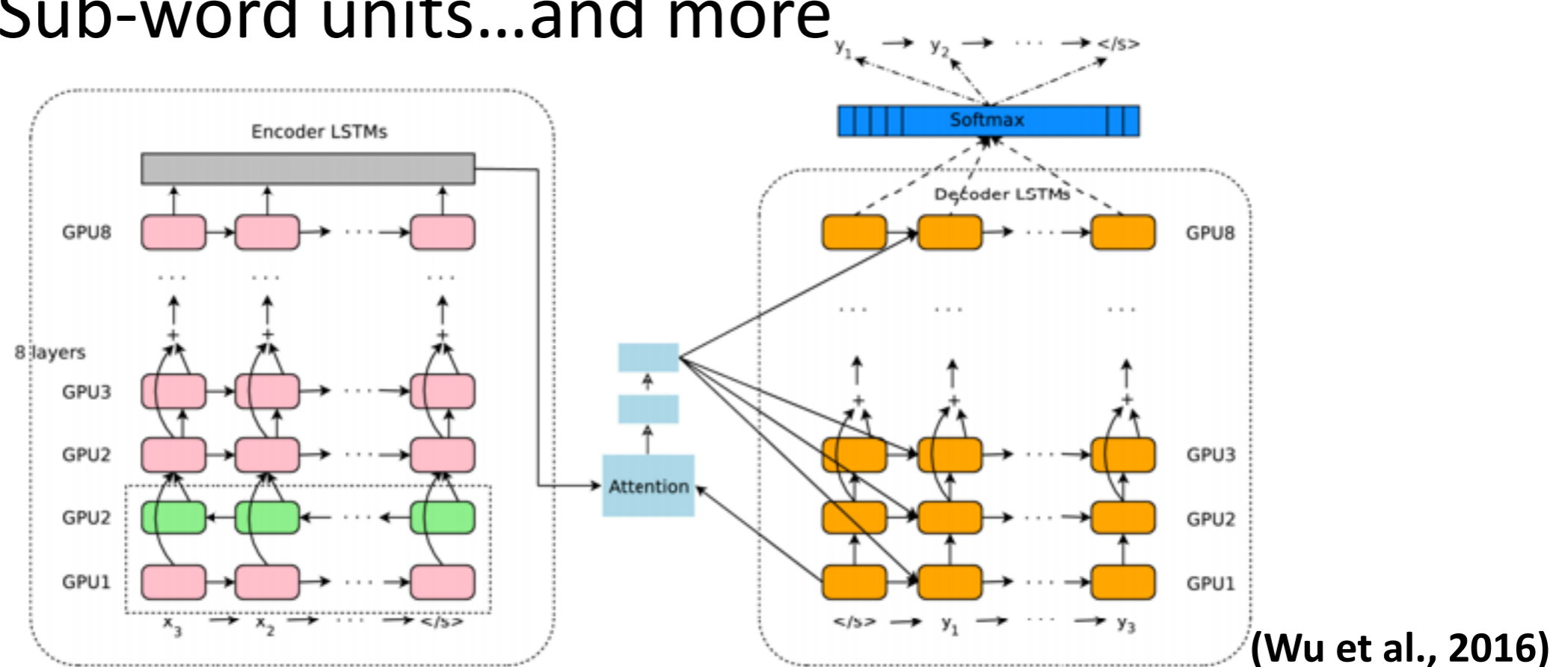
			Fr (39m)		Cs (12m)		De (4.2m)		Ru (2.3m)		Fi (2m)	
			→ En	En →	→ En	En →	→ En	En →	→ En	En →	→ En	En →
(a) BLEU	Dev	Single	27.22	26.91	21.24	15.9	24.13	20.49	21.04	18.06	13.15	9.59
		Multi	26.09	25.04	21.23	14.42	23.66	19.17	21.48	17.89	12.97	8.92
	Test	Single	27.94	<b>29.7</b>	20.32	<b>13.84</b>	24	<b>21.75</b>	22.44	<b>19.54</b>	12.24	<b>9.23</b>
		Multi	<b>28.06</b>	27.88	<b>20.57</b>	13.29	<b>24.20</b>	20.59	<b>23.44</b>	19.39	<b>12.61</b>	8.98
(b) LL	Dev	Single	-50.53	-53.38	-60.69	-69.56	-54.76	-61.21	-60.19	-65.81	-88.44	-91.75
		Multi	-50.6	-56.55	-54.46	-70.76	-54.14	-62.34	-54.09	-63.75	-74.84	-88.02
	Test	Single	-43.34	<b>-45.07</b>	-60.03	<b>-64.34</b>	-57.81	<b>-59.55</b>	-60.65	-60.29	-88.66	-94.23
		Multi	<b>-42.22</b>	-46.29	<b>-54.66</b>	-64.80	<b>-53.85</b>	-60.23	<b>-54.49</b>	<b>-58.63</b>	<b>-71.26</b>	<b>-88.09</b>

(Firat et al., 2016)



# Google's Neural Machine Translation System

- An encoder, a decoder and an attention network
  - 8-layer deep with residual connections
  - Refinement with Reinforcement Learning
  - Sub-word units...and more



# Google's Neural Machine Translation System

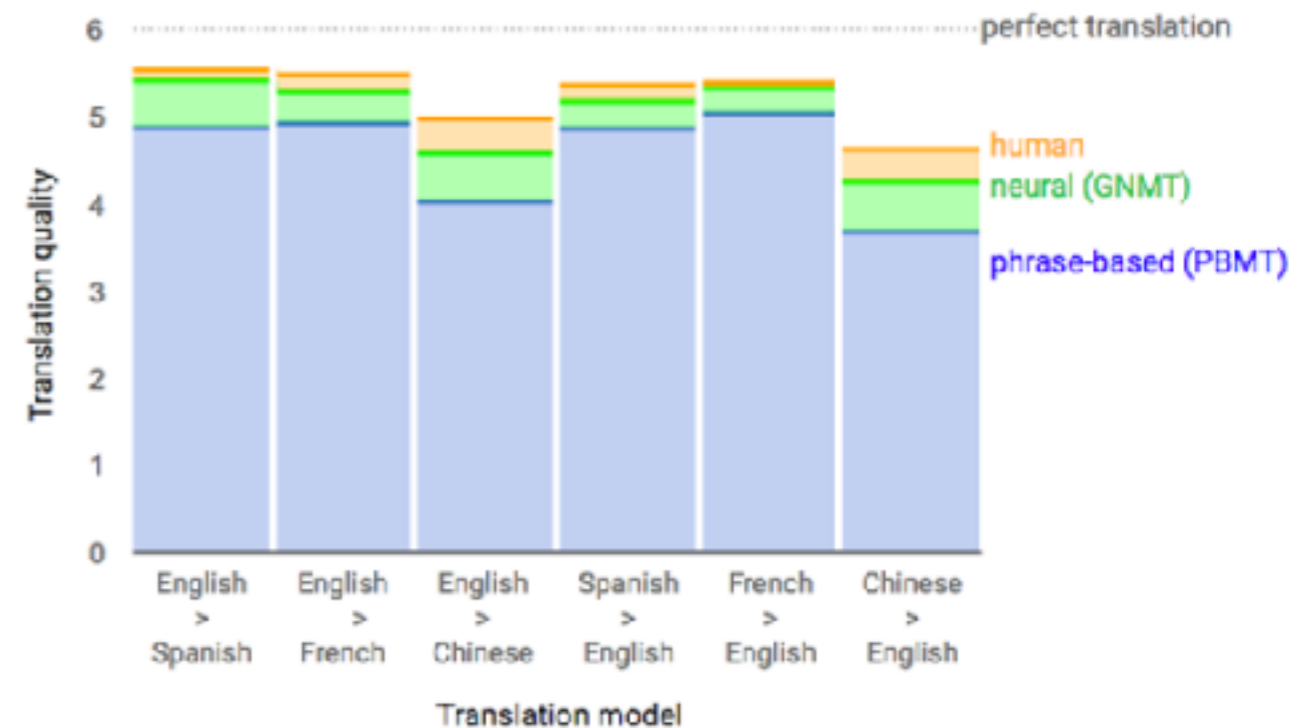
- EN->FR training took 6 days on 96GPUS !!!! and 3 more days for refinement...

Table 7: Model ensemble results on WMT En→Fr (newstest2014)

Model	BLEU
WPM-32K (8 models)	40.35
RL-refined WPM-32K (8 models)	41.16
LSTM (6 layers) [31]	35.6
LSTM (6 layers + PosUnk) [31]	37.5
Deep-Att + PosUnk (8 models) [45]	40.4

Table 5: Single model results on WMT En→De (newstest2014)

Model	BLEU	CPU decoding time per sentence (s)
Word	23.12	0.2972
Character (512 nodes)	22.62	0.8011
WPM-8K	23.50	0.2079
WPM-16K	24.36	0.1931
WPM-32K	24.61	0.1882
Mixed Word/Character	24.17	0.3268
PBMT [6]	20.7	
RNNSearch [37]	16.5	
RNNSearch-LV [37]	16.9	
RNNSearch-LV [37]	16.9	
Deep-Att [45]	20.6	



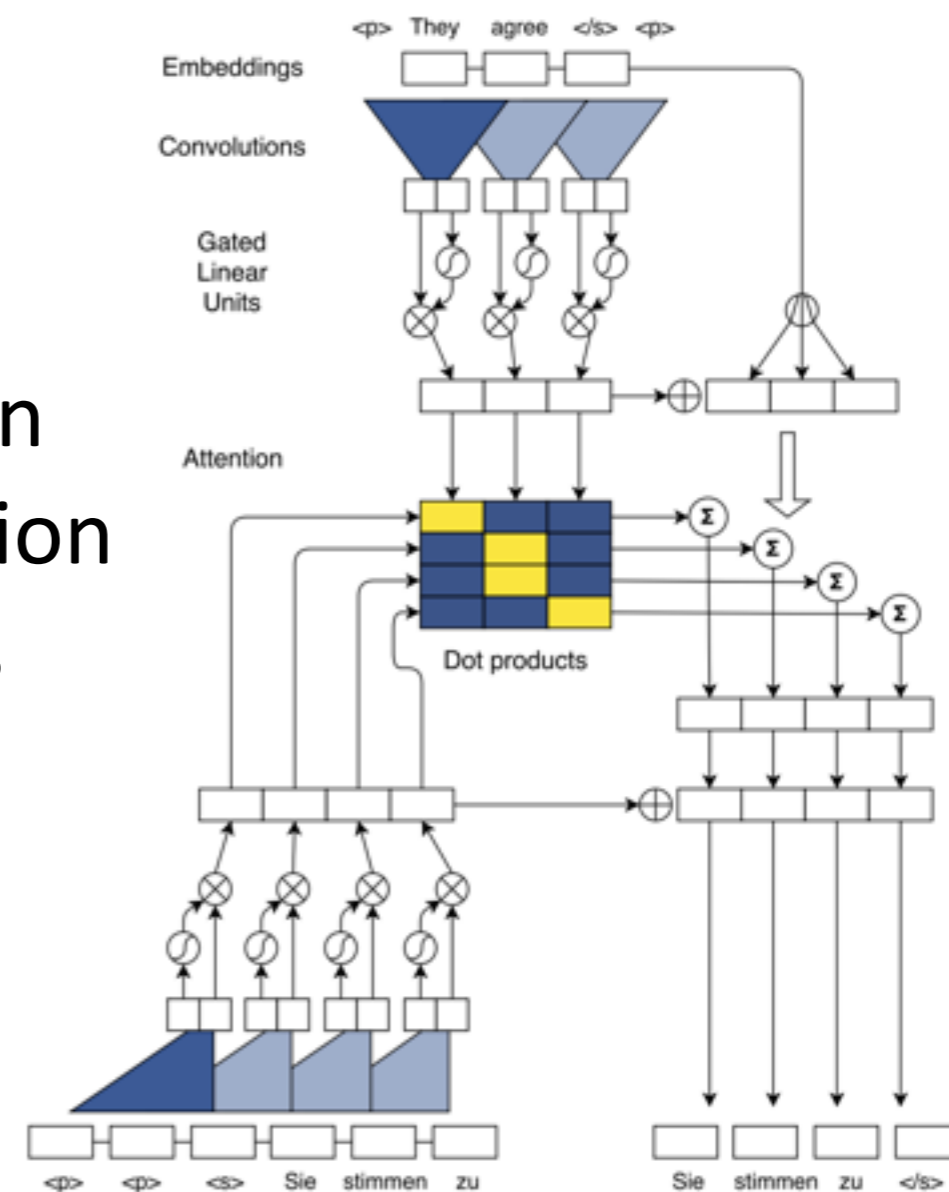
Data from side-by-side evaluations, where human raters compare the quality of translations for a given source sentence. Scores range from 0 to 6, with 0 meaning "completely nonsense translation", and 6 meaning "perfect translation."

(Wu et al., 2016)

# Convolutional Encoder-Decoder

- Outperformed GNMT and was more efficient in terms of speed, but
  - Lacks long-term memory
  - Requires meticulous initialization schemes and careful normalization
  - Requires positional embeddings
  - Requires more depth (15 layers)

WMT'14 English-German	BLEU
Luong et al. (2015) LSTM (Word 50K)	20.9
Kalchbrenner et al. (2016) ByteNet (Char)	23.75
Wu et al. (2016) GNMT (Word 80K)	23.12
Wu et al. (2016) GNMT (Word pieces)	24.61
ConvS2S (BPE 40K)	25.16



(Gehring et al., 2017)

# Self-Attention or Transformer Networks

- Encode/Decode input w.o. using CNNs or LSTMs
- Lower training cost but lack long-term memory
- Stacked self-attention with multiple heads

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

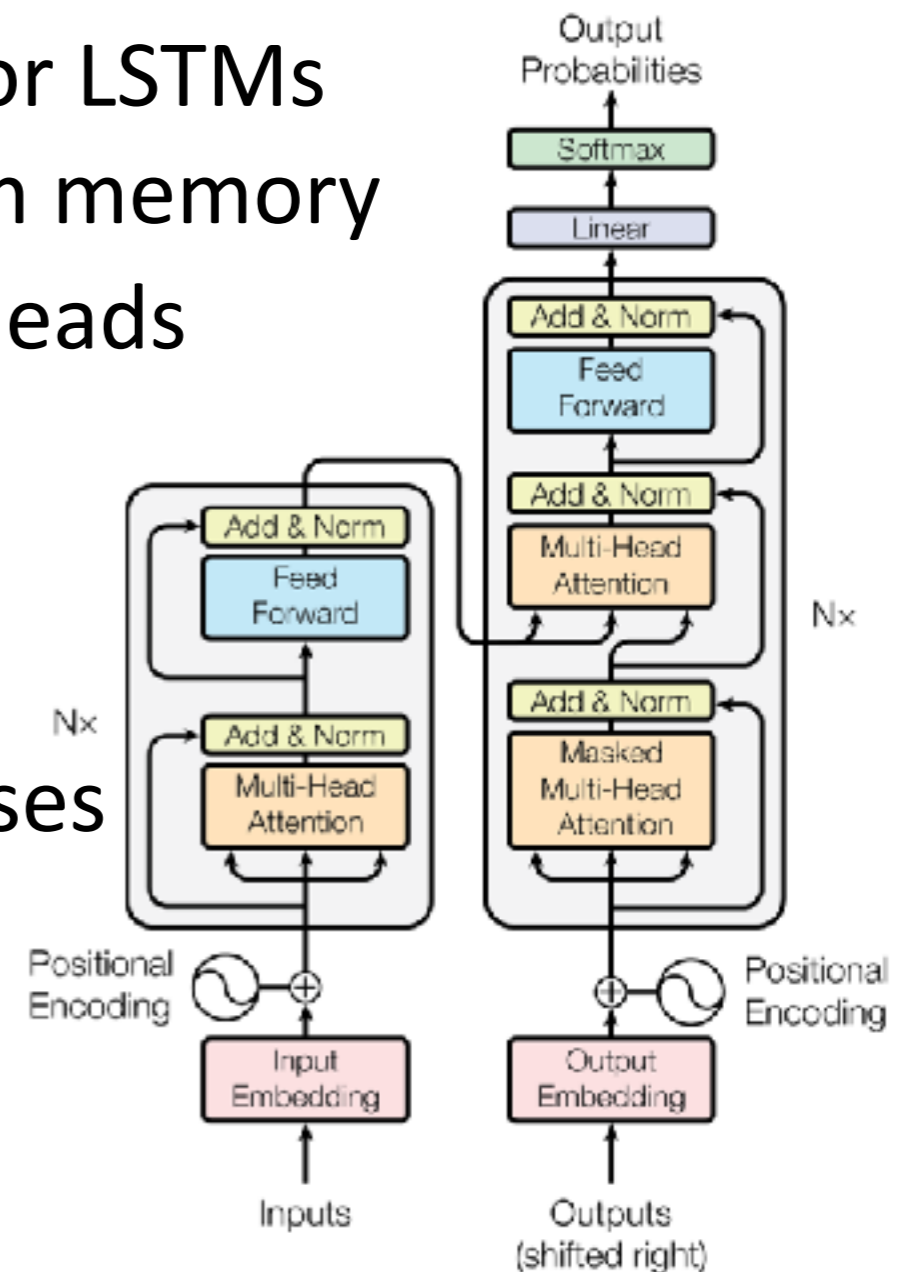
$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \dots, \text{head}_h)W^O$$

where  $\text{head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$

- To capture sequence information it uses positional embeddings (sinusoids)

$$PE_{(pos, 2i)} = \sin(pos/10000^{2i/d_{\text{model}}})$$

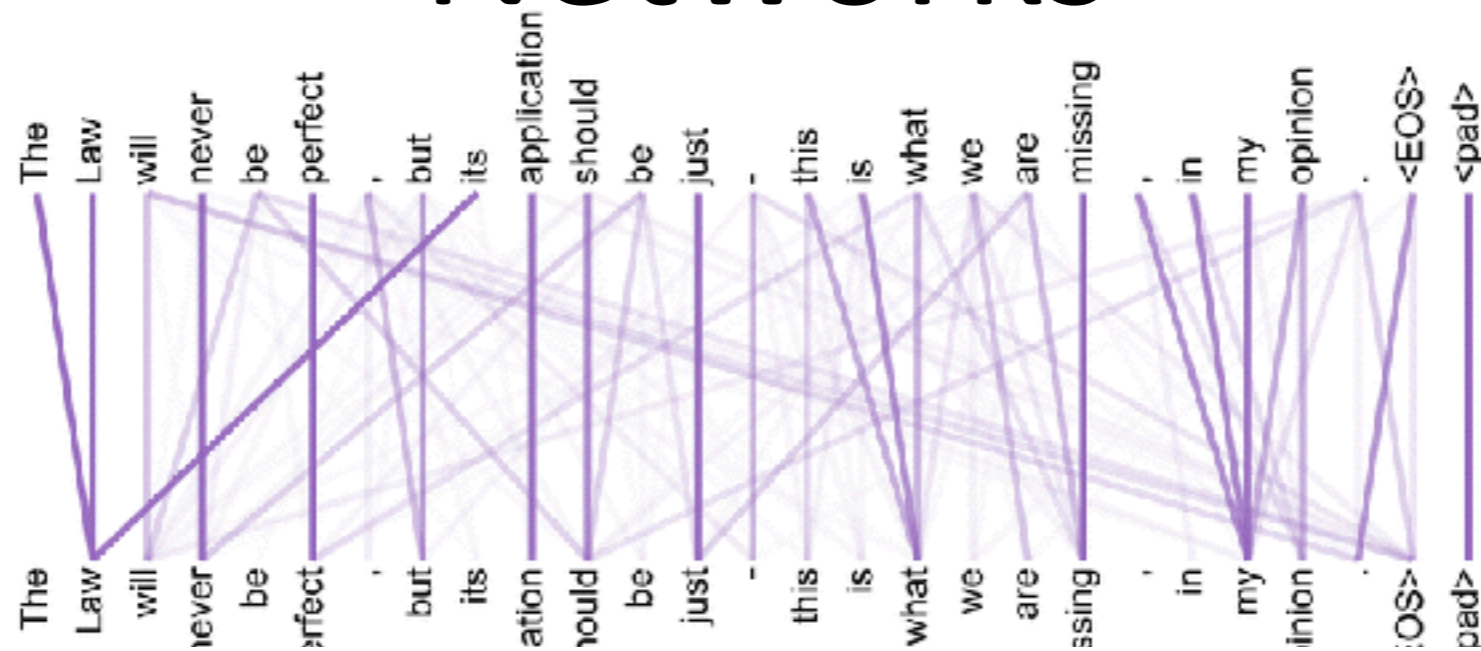
$$PE_{(pos, 2i+1)} = \cos(pos/10000^{2i/d_{\text{model}}})$$



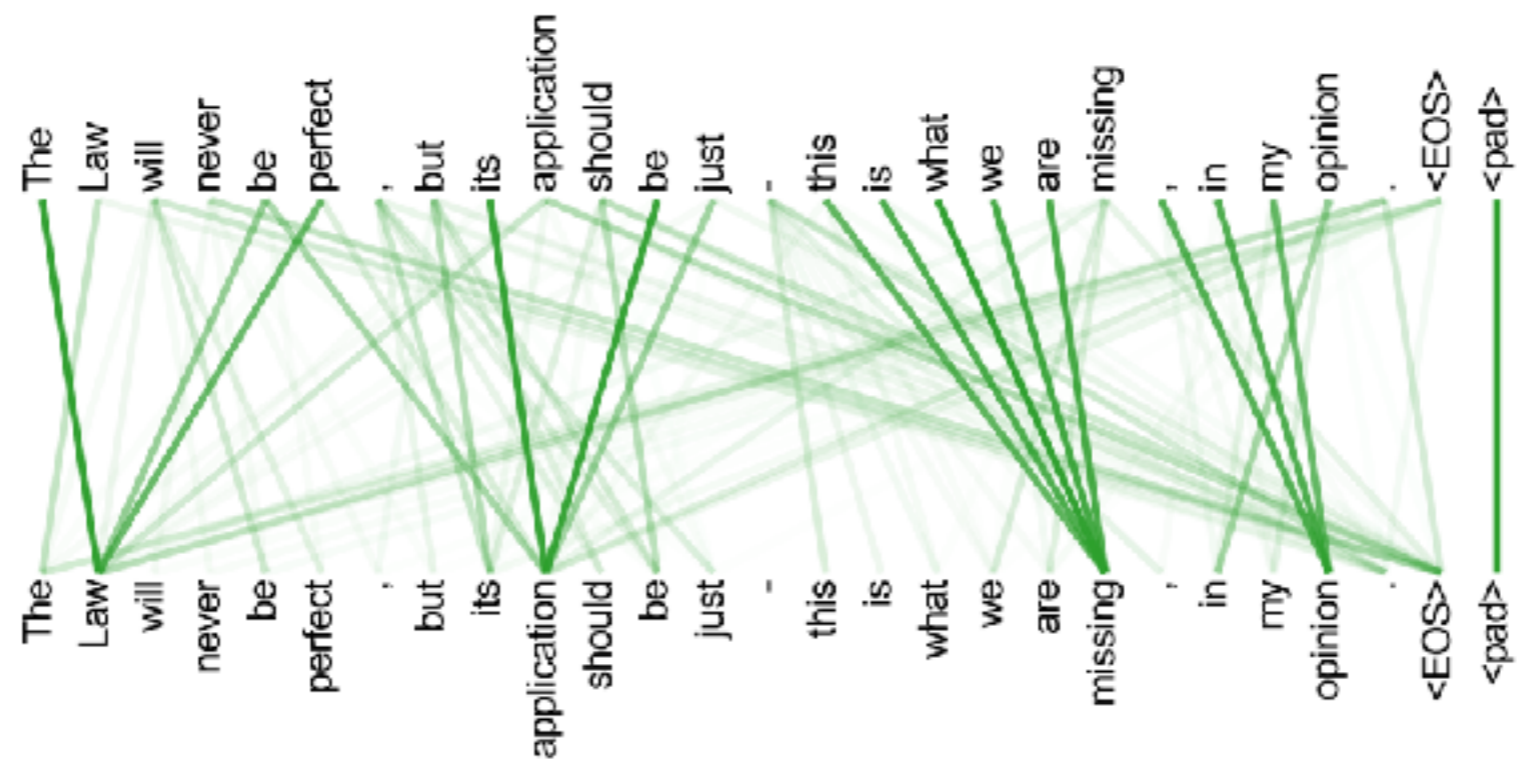


# Self-Attention or Transformer Networks

Head 1

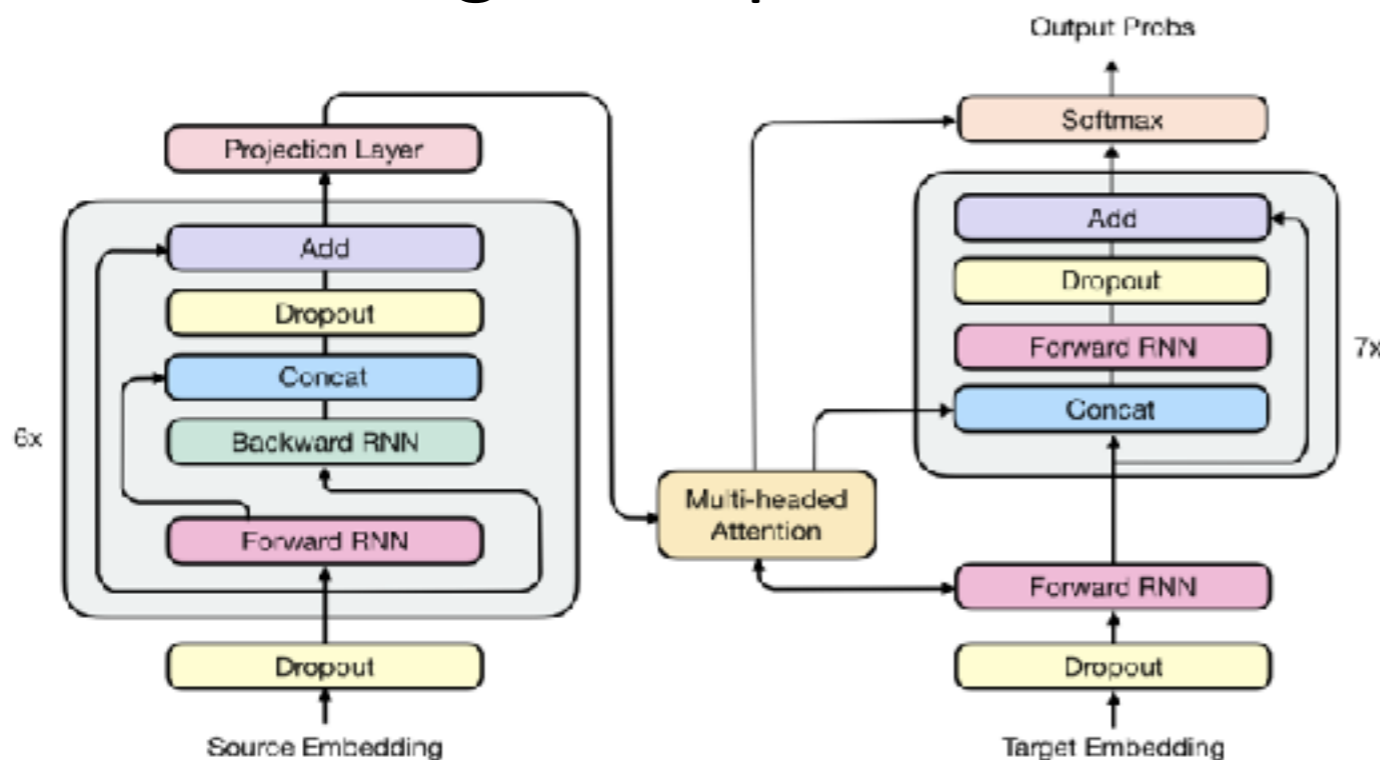


Head 2



# Best of both worlds: LSTMs & Transformer Networks

- Transformer network uses techniques that other models do not use
  - Combine strengths of both worlds
  - CNNs fall behind in BLEU and convergence speed



Model	Test BLEU	Epochs	Training Time
GNMT	38.95	-	-
ConvS2S <sup>7</sup>	39.49 ± 0.11	62.2	438h
Trans. Base	39.43 ± 0.17	20.7	90h
Trans. Big <sup>8</sup>	40.73 ± 0.19	8.3	120h
RNMT+	41.00 ± 0.05	8.5	120h

Table 1: Results on WMT14 En→Fr.

Model	Test BLEU	Epochs	Training Time
GNMT	24.67	-	-
ConvS2S	25.01 ± 0.17	38	20h
Trans. Base	27.26 ± 0.15	38	17h
Trans. Big	27.94 ± 0.18	26.9	48h
RNMT+	28.49 ± 0.05	24.6	40h

Table 2: Results on WMT14 En→De.

Encoder	Decoder	En→Fr Test BLEU
Trans. Big	Trans. Big	40.73 ± 0.19
RNMT+	RNMT+	41.00 ± 0.05
Trans. Big	RNMT+	<b>41.12 ± 0.16</b>
RNMT+	Trans. Big	39.92 ± 0.21

Table 5: Results for encoder-decoder hybrids.

(Chen & Firat & Bapna et al., 2018)



# Outline of the talk

1. Introduction and Motivation
2. Word Representation Learning
  - Semantic similarity
  - Traditional and recent approaches
  - Intrinsic and extrinsic evaluation
3. Word Sequence Modeling
  - Essentials: RNNs, Attention, DL tricks
  - Text Classification
  - Machine Translation
4. Conclusion and Discussion

# Conclusion

- Deep learning for NLP has flourished the last couple of years
  - ➔ Attention mechanism became popular even outside NLP
  - ➔ Companies heavily use NLP e.g. machine translation
- Although attention-based models can get us far
  - ➔ Linguistic structure can get us even further by constraining models and creating useful inductive biases
  - ➔ Inspecting and analyzing the learned structures can help us gain insights about language
  - ➔ There is still a lot of work to be done on weakly-supervised and unsupervised learning

# Future Challenges

- Transferring knowledge across domains, languages and different outputs
- Contextualizing word representations
- Generalizing on unseen examples
- Learning from very few examples
- Summarization / Entailment / Reasoning

# Discussion / Coding Session

- Learning word embeddings  
<https://www.tensorflow.org/tutorials/word2vec>  
<https://nlp.stanford.edu/projects/glove/>
- Classification using pre-trained word embeddings  
<https://blog.keras.io/using-pre-trained-word-embeddings-in-a-keras-model.html>



- ➔ <https://www.tensorflow.org/install/>
- ➔ <https://keras.io/#installation>
- ➔ <https://keras.io/backend/>