Learning Deep Learning on example of Seq2seq with attention

by Andriy Gryshchuk

senior research engineer
Grammarly
https://www.linkedin.com/in/andreygryshchuk/
https://www.kaggle.com/anmiko

Outline

- DL vs. ML
  - ML approach
  - DL approaches
    - FCN
    - CNN
    - RNN
- Seq2Seq
  - Decoder
  - Encoder
  - Attention
- Seq2Seq applications

Intro

Disclaimers

You cannot learn DL by listening to lectures

Why lectures are not good?

To learn Deep Learning

Why lectures are not good?

To learn Deep Learning

Definition

Theory

Empiricism

Trial and error

Why lectures are not good?

Trials and errors

Huge search space

Heuristics

Intuition

Example task - GED

X	Y
She win a song’s contest.	1
He did not win the contest.	0
...	..

Classical ML

Classical ML

- Features
-

Classical ML

- Features
- Features
- Features
- Algorithms

Classical ML

90% efforts creating features

10% efforts modelling

DL big promise

DL big promise

The model will learn good ‘*features*’ on its own from raw inputs

Feature engineering - no more!

Domain knowledge? - not required

DL big promise

The model will learn good '*features*' (internal representations) on its own from *raw* inputs

raw inputs:

- pixels
- words
- characters
- sound waves
- ...

Example task - GED

X	Y
She win a song’s contest.	1
He did not win the contest.	0
...	..

How to approach with DL?

How to approach with DL?

Fully Connected Network?

How to approach with DL?

Fully Connected Network - why not

Text to vectors?

One-hot encoding

Problems?

One-hot encoding

Problems

- Sparsity
- No meaningful distance

Dense vectors aka Embeddings

Tip

- Use pre-trained embeddings and freeze them if you are data poor
- Use pre-trained embeddings and train them if you are not that poor
- Train from scratch if you are data rich
  - Still think about pre-training your own

FCN

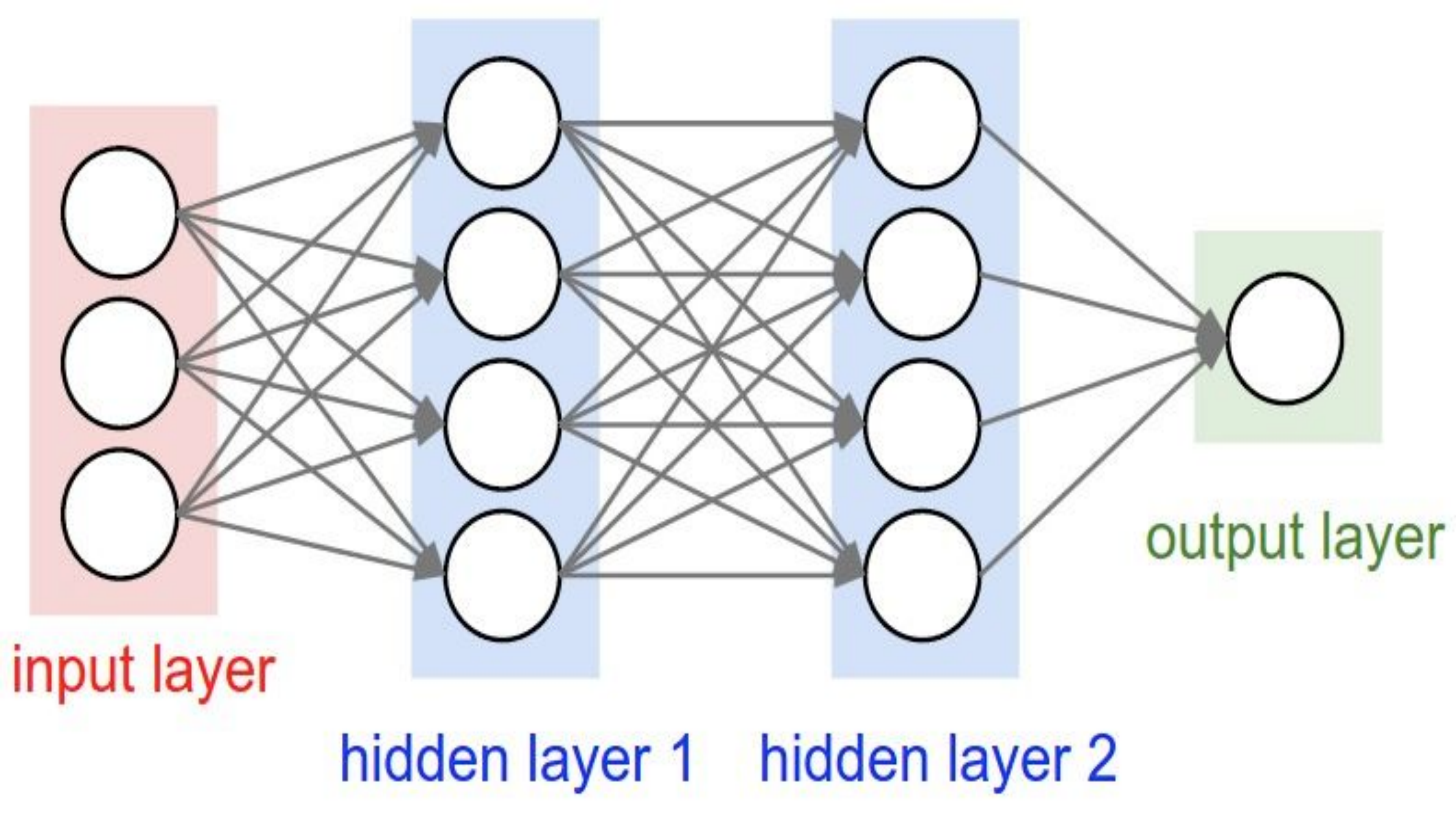
Good or not?

FCN

Good or not?

Theory - you can approximate any function with a FCN

Practice - prone to overfitting, hard to train, too many parameters


input layer
hidden layer 1
hidden layer 2
output layer

FCN

Good or not?

- Sentence length <=30
- Embeddings dim - 300
- The size of the first hidden layer - 200
- The size of the weight matrix?

FCN

Good or not?

- Sentence length: 30
- Embeddings dim: 300
- Input layer: 30x300
- The size of the first hidden layer: 200
- The size of the weight matrix : 9000x200

FCN

Good or not?

- Used as a part of more complex architectures
- Still useful when you have engineered features
-

Restrict them

Inductive Bias or prior knowledge about nature of the problem we try to solve

N-grams - n-previous dependence

CNN - local connectivity, shared weights for filters

RNN - sequential nature of the problem

CNN

Convolution networks for text?

CNN

Convolution networks for text?

Sure

1D convolutions

CNNs

- Much faster than RNN
- Similar or better accuracy
- N-grams on steroids
- Replacing RNN in many domains

Tip

Consider CNN as the first choice - allows faster iterations

CNN

Tip

Look at what image (other) folks are doing

RNNs are not dead

- Widely used in NLP
- Natural choice to work with sequences

Example task - GED

X	Y
She win a song’s contest.	1
He did not win the contest.	0
...	..

Input Tokens?

Input Tokens

- characters
- words
- subword units (BPE and others)

Sequence to Sequence models

Example task - GED

X	Y
She win a song’s contest.	1
He did not win the contest.	0
...	..

POS tagging

The cat sit on the mat => DT NN VB IN DT NN

NLP as sequence to sequence

The cat sit on the mat => Error

The cat sit on the mat => DT NN VB IN DT NN

The cat sit on the mat => The cat {sit=>sits} on the mat

, What color is the mat? => Red

Machine Translation

She is going to visit Paris => Вона збирається відвідати Париж

Neural Machine Translation

- wide applications
- rapid progress
- data
- top area of research

Neural Machine Translation

She is going to visit Paris => Вона збирається відвідати Париж

Given sentence pairs (source, target) train a model which will translate sentences in one language to another

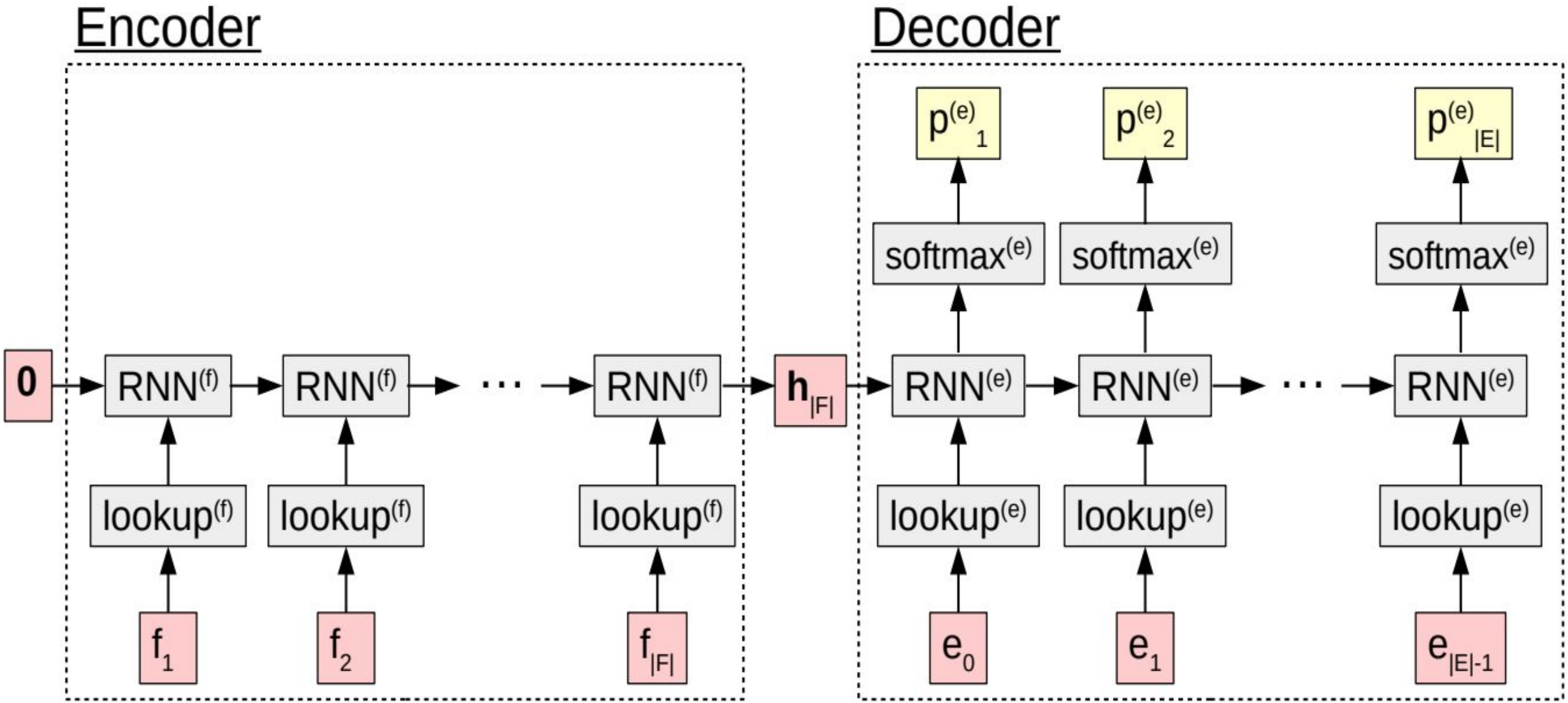


image from https://arxiv.org/pdf/1703.01619v1.pdf

Decoder

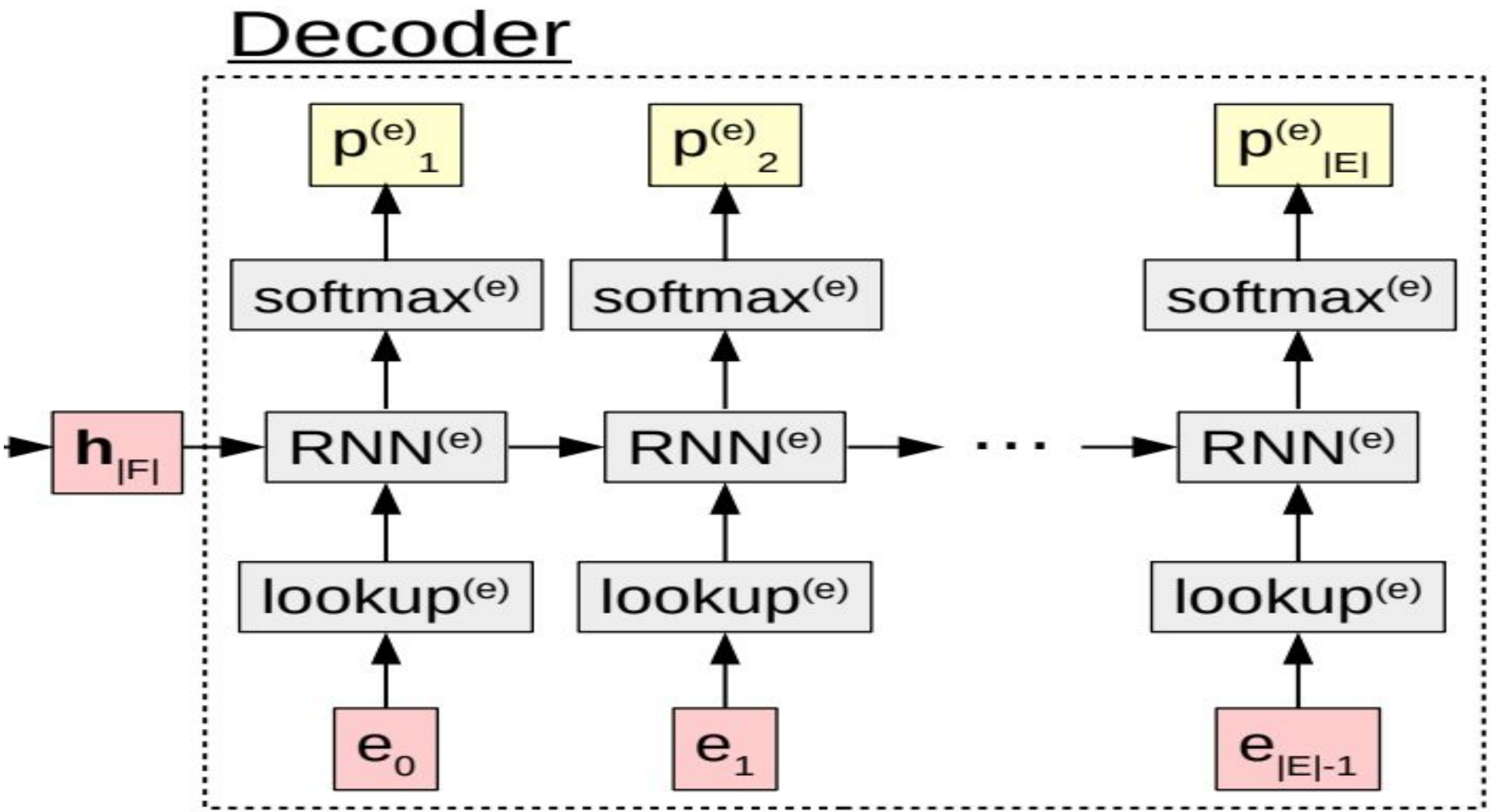


image from https://arxiv.org/pdf/1703.01619v1.pdf

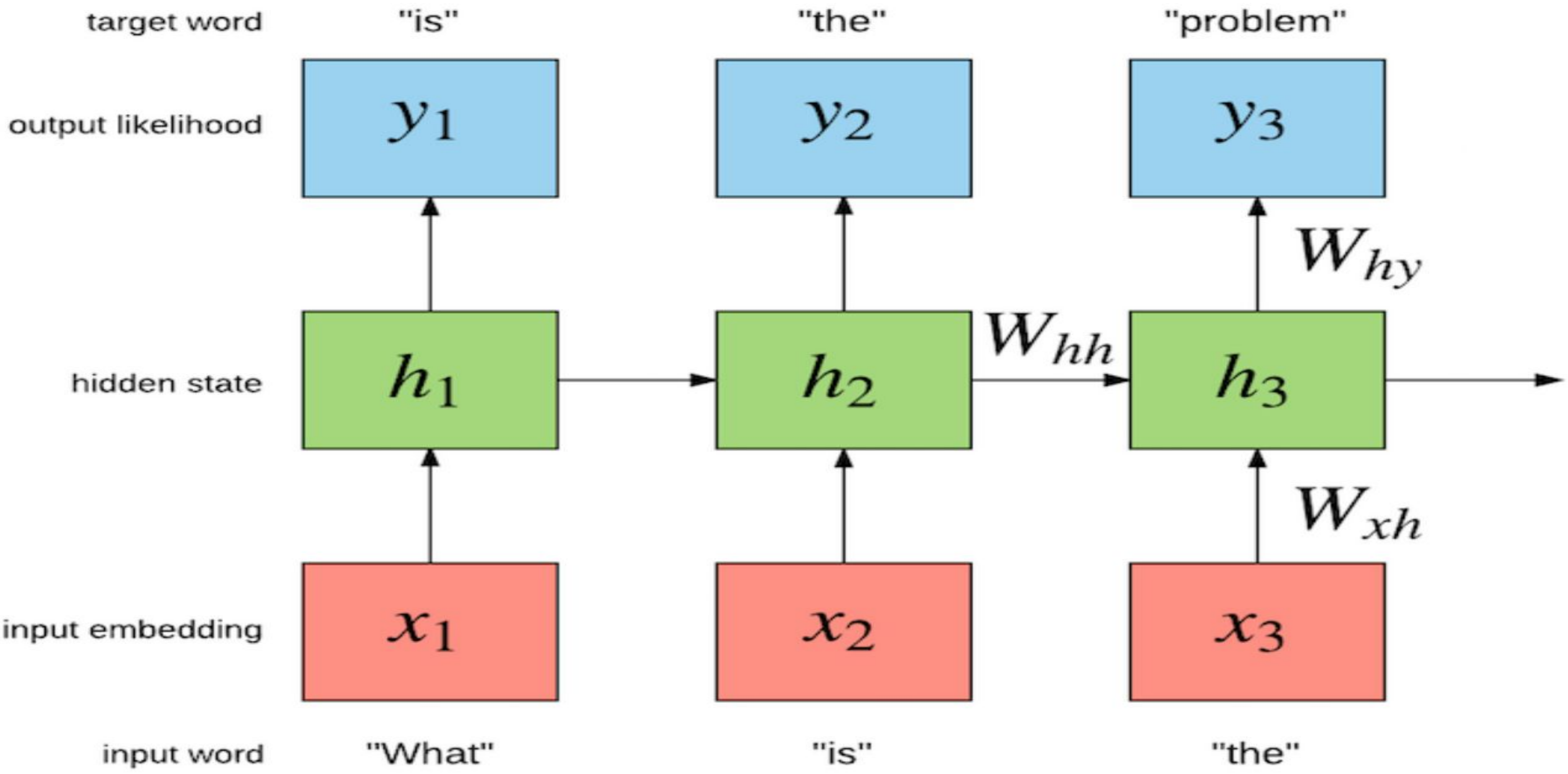


image from http://torch.ch/blog/2016/07/25/nce.html

Decoder

Just a language model with non-zero initial hidden state

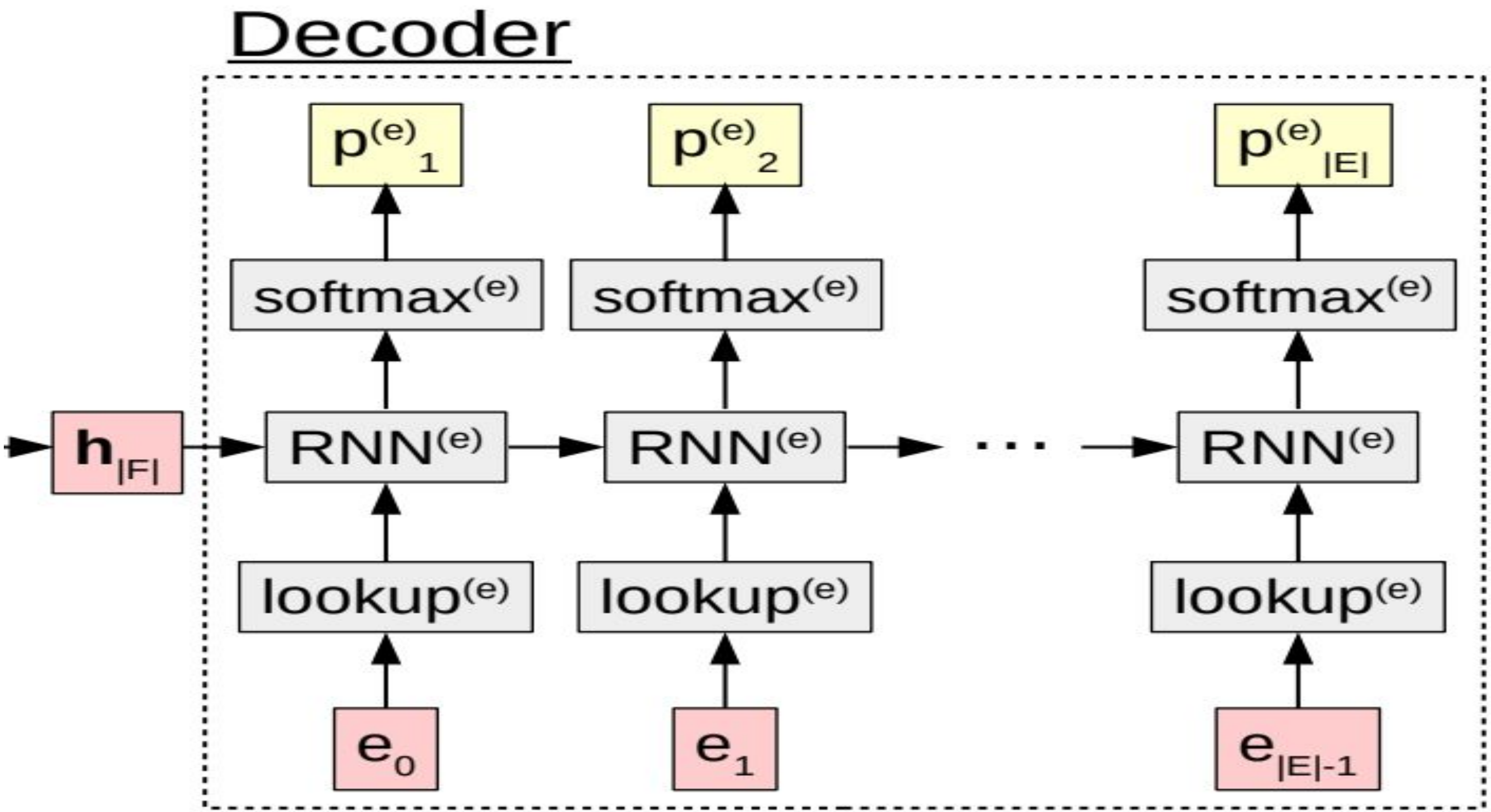


image from https://arxiv.org/pdf/1703.01619v1.pdf

How to generate output sentence

Autoregressive!

Each time step we get a probability distribution over our vocabulary given already generated tokens

How to generate output sentence

Autoregressive!

Each time step we get a probability distribution over our vocabulary given already generated tokens

Greedy - is suboptimal

Beam search - better but more expensive

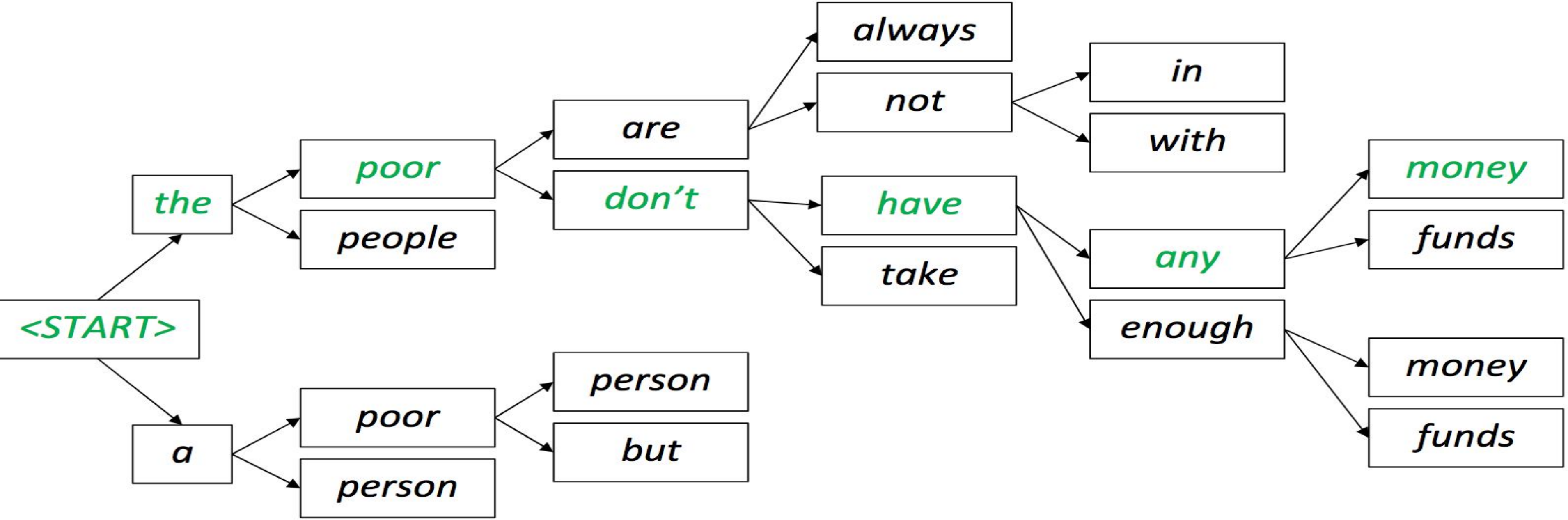
- Greedy decoding has no way to undo decisions!
  - *les pauvres sont démunis (the poor don't have any money)*
  - → *the* ____
  - → *the poor* ____
  - → *the poor are* ____

- Better option: use beam search (a search algorithm) to explore *several* hypotheses and select the best one

slide from http://web.stanford.edu/class/cs224n/syllabus.html

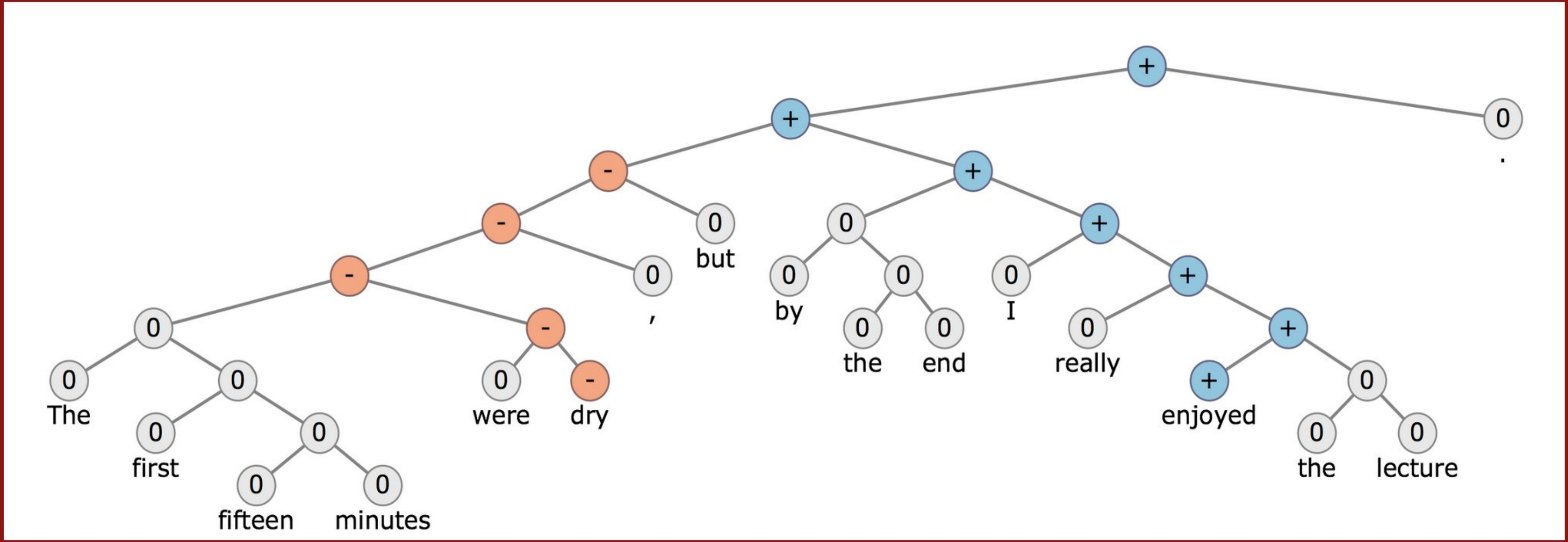
Beam search decoding: example

Beam size = 2



slide from http://web.stanford.edu/class/cs224n/syllabus.html

Archived: CS224n: Natural Language Processing with Deep Learning (2016-2017)



Syllabus (Slides, etc.) | Public Lecture Videos | Stanford Lecture Videos | Piazza forum

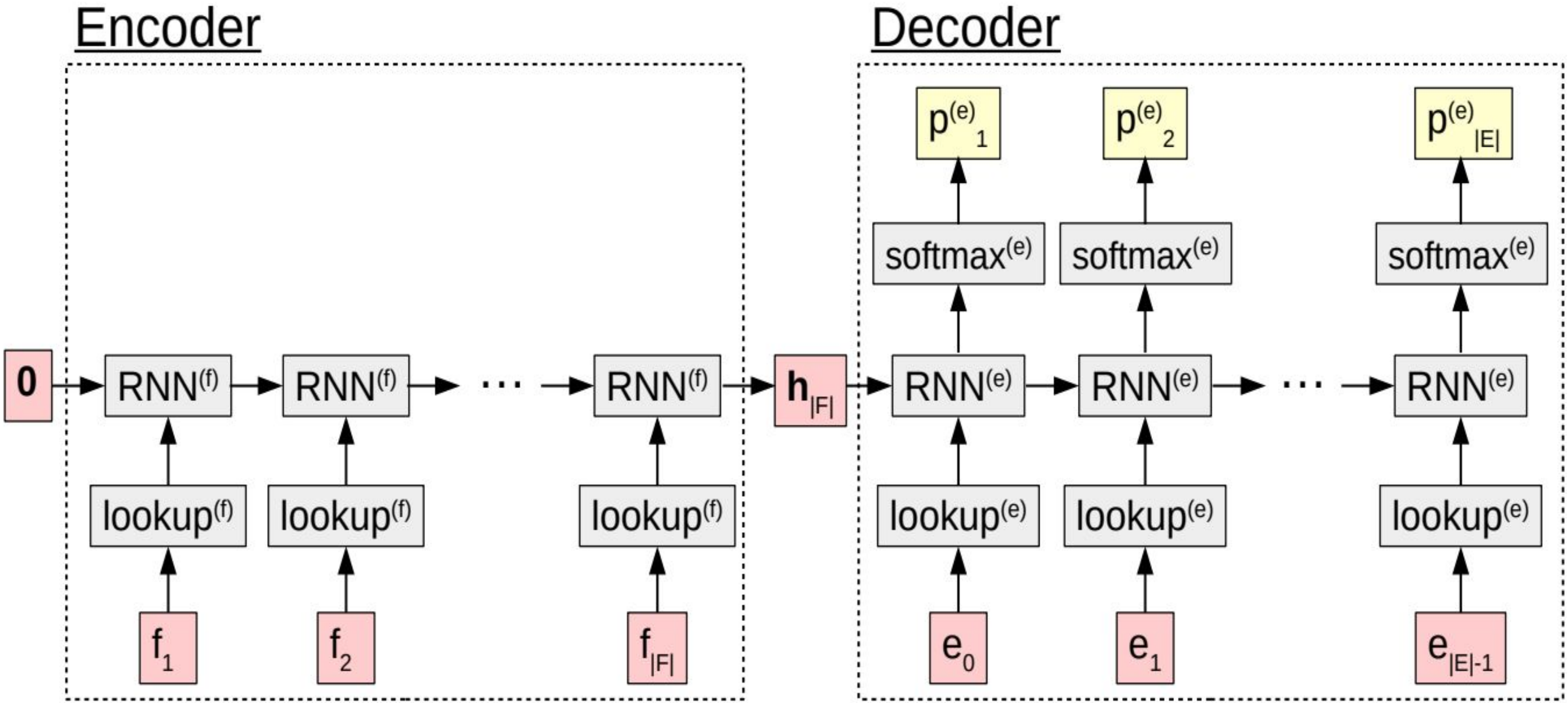


image from https://arxiv.org/pdf/1703.01619v1.pdf

Encoder

Plenty of options

- bag of words
- RNN
- CNN
- Tree network

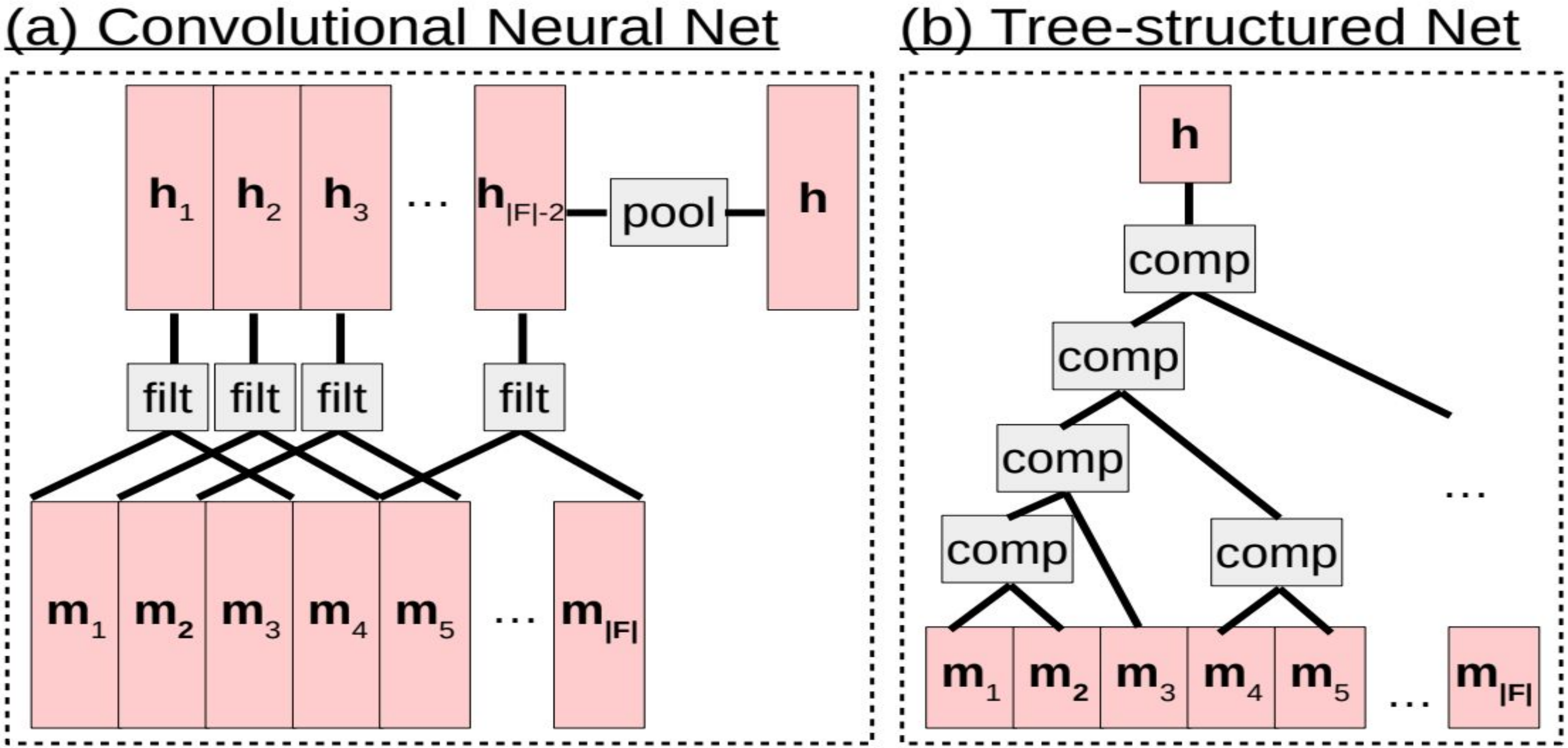


image from https://arxiv.org/pdf/1703.01619v1.pdf

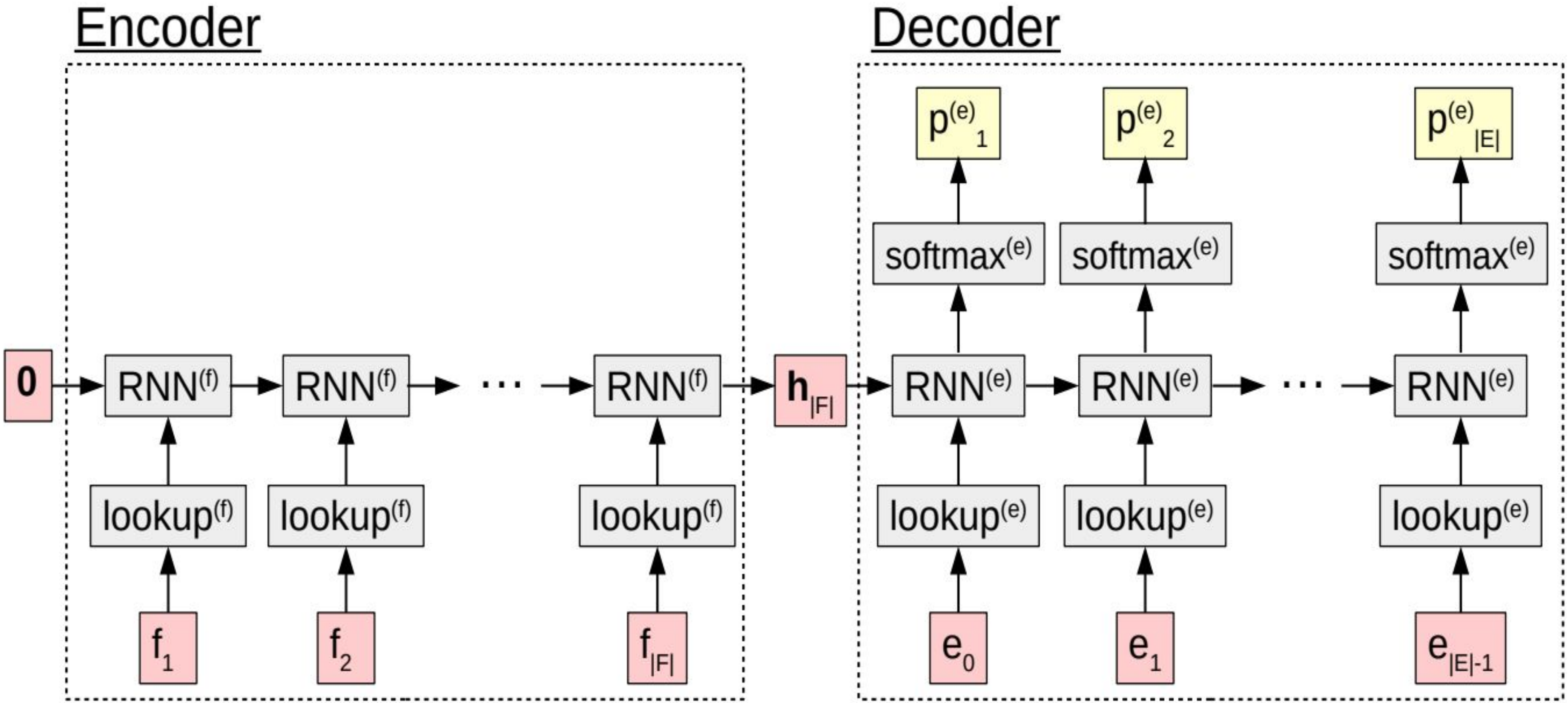


image from https://arxiv.org/pdf/1703.01619v1.pdf

Encoder and Decoder are trained together

Could be pre-trained separately
For example with LM target

Embedding could be pretrained as well - usually trained from scratch

Encoder - Decoder problems

output hidden state bottleneck

long term dependencies

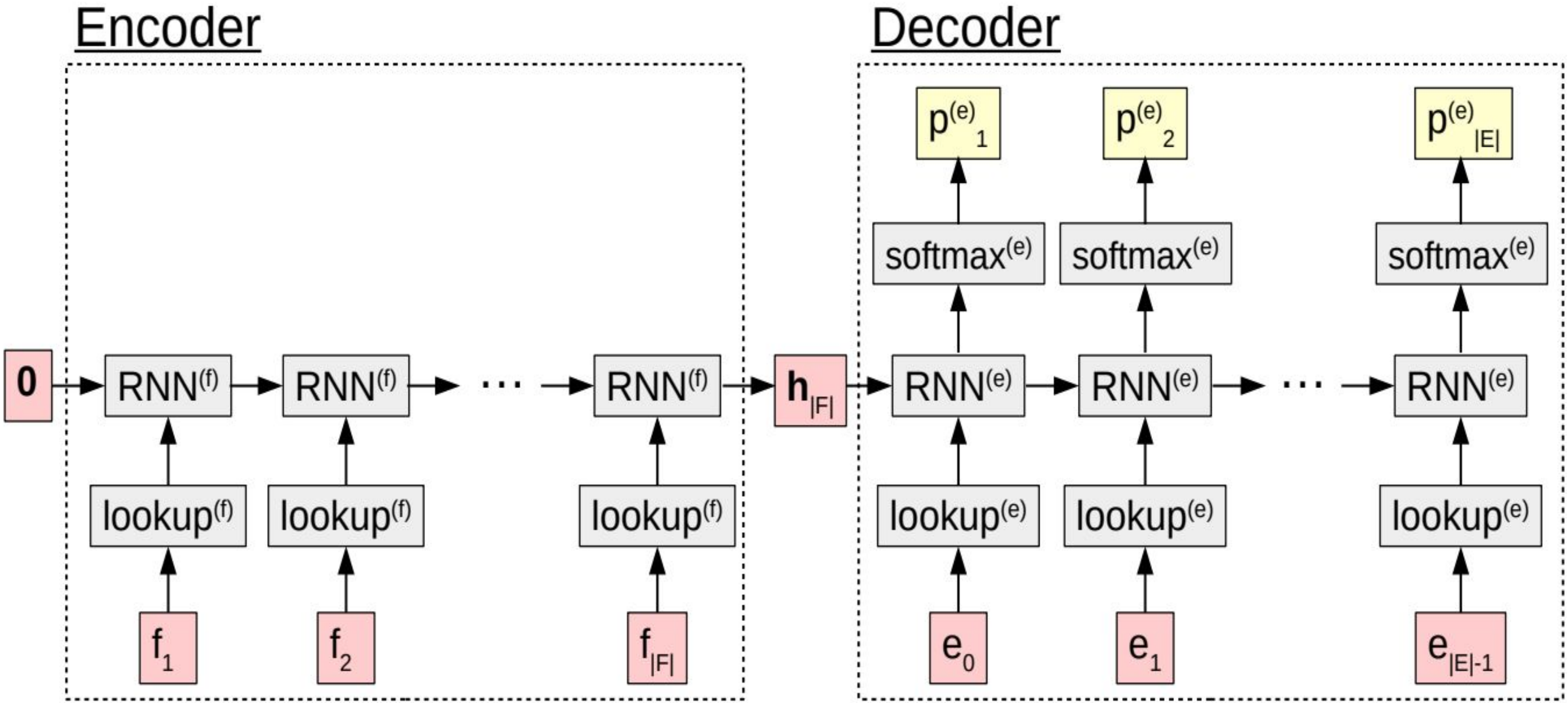


image from https://arxiv.org/pdf/1703.01619v1.pdf

Attention Revolution

arXiv.org > cs > arXiv:1409.0473

Search or Articl

(Help | Advanced sea

Computer Science > Computation and Language

Neural Machine Translation by Jointly Learning to Align and Translate

Dzmitry Bahdanau, Kyunghyun Cho, Yoshua Bengio

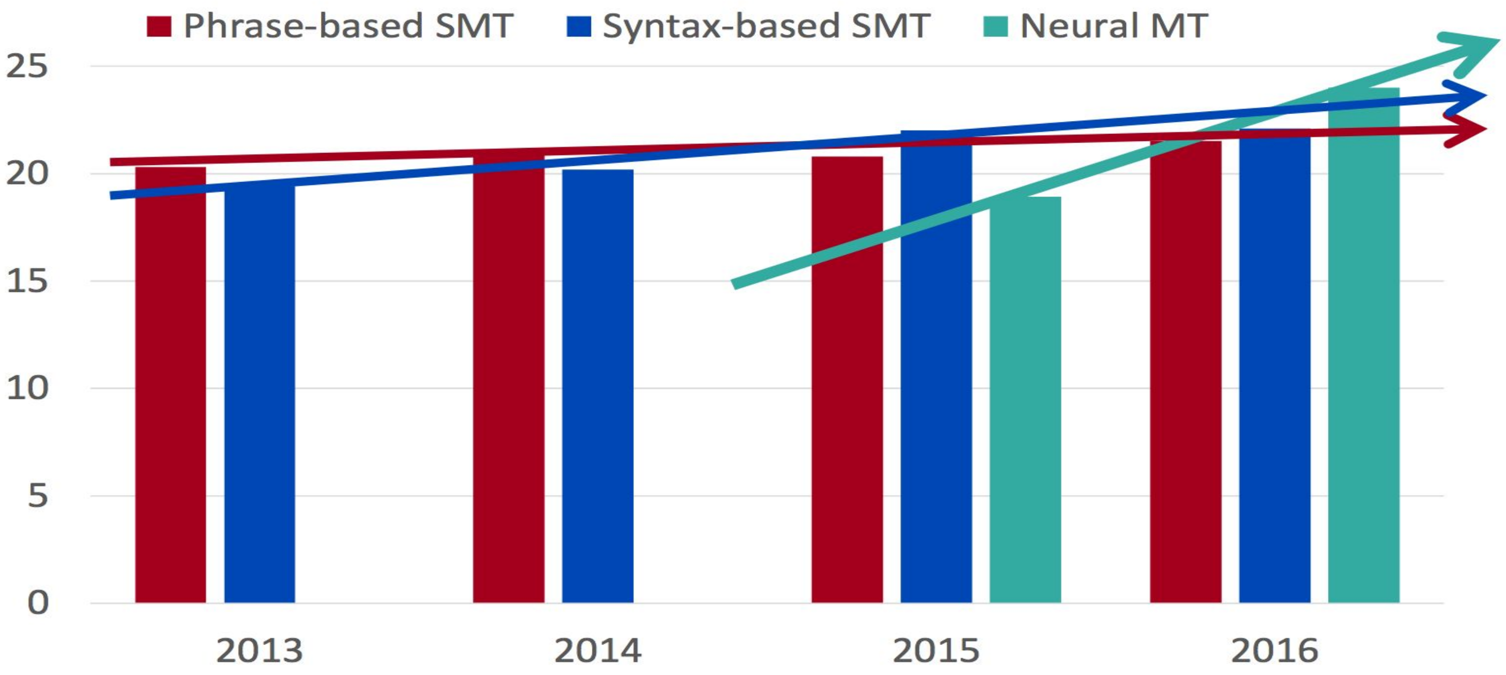
(Submitted on 1 Sep 2014 (v1), last revised 19 May 2016 (this version, v7))

Neural machine translation is a recently proposed approach to machine translation. Unlike the traditional statistical machine translation, the neural machine translation aims at building a single neural network that can be jointly tuned to maximize the translation performance. The models proposed recently for neural machine translation often belong to a family of encoder-decoders and consists of an encoder that encodes a source sentence into a fixed-length vector from which a decoder generates a translation. In this paper, we conjecture that the use of a fixed-length vector is a bottleneck in improving the performance of this basic encoder-decoder architecture, and propose to extend this by allowing a model to automatically (soft-)search for parts of a source sentence that are relevant to predicting a target word, without having to form these parts as a hard segment explicitly. With this new approach, we achieve a translation performance comparable to the existing state-of-the-art phrase-based system on the task of English-to-French translation. Furthermore, qualitative analysis reveals that the (soft-)alignments found by the model agree well with our intuition.

Comments:	Accepted at ICLR 2015 as oral presentation
Subjects:	**Computation and Language (cs.CL)**; Learning (cs.LG); Neural and Evolutionary Computing (cs.NE); Machine Learning (stat.ML)
Cite as:	arXiv:1409.0473 **[cs.CL]**
	(or arXiv:1409.0473v7 **[cs.CL]** for this version)

MT progress over time

[Edinburgh En-De WMT newstest2013 Cased BLEU; NMT 2015 from U. Montréal]



Source: http://www.meta-net.eu/events/meta-forum-2016/slides/09_sennrich.pdf

slide from http://web.stanford.edu/class/cs224n/syllabus.html

NMT: the biggest success story of NLP Deep Learning

Neural Machine Translation went from a fringe research activity in **2014** to the leading standard method in **2016**

- **2014**: First seq2seq paper published
- **2016**: Google Translate switches from SMT to NMT
- This is amazing!
  - **SMT** systems, built by hundreds of engineers over many years, outperformed by NMT systems trained by a handful of engineers in a few months

slide from http://web.stanford.edu/class/cs224n/syllabus.html

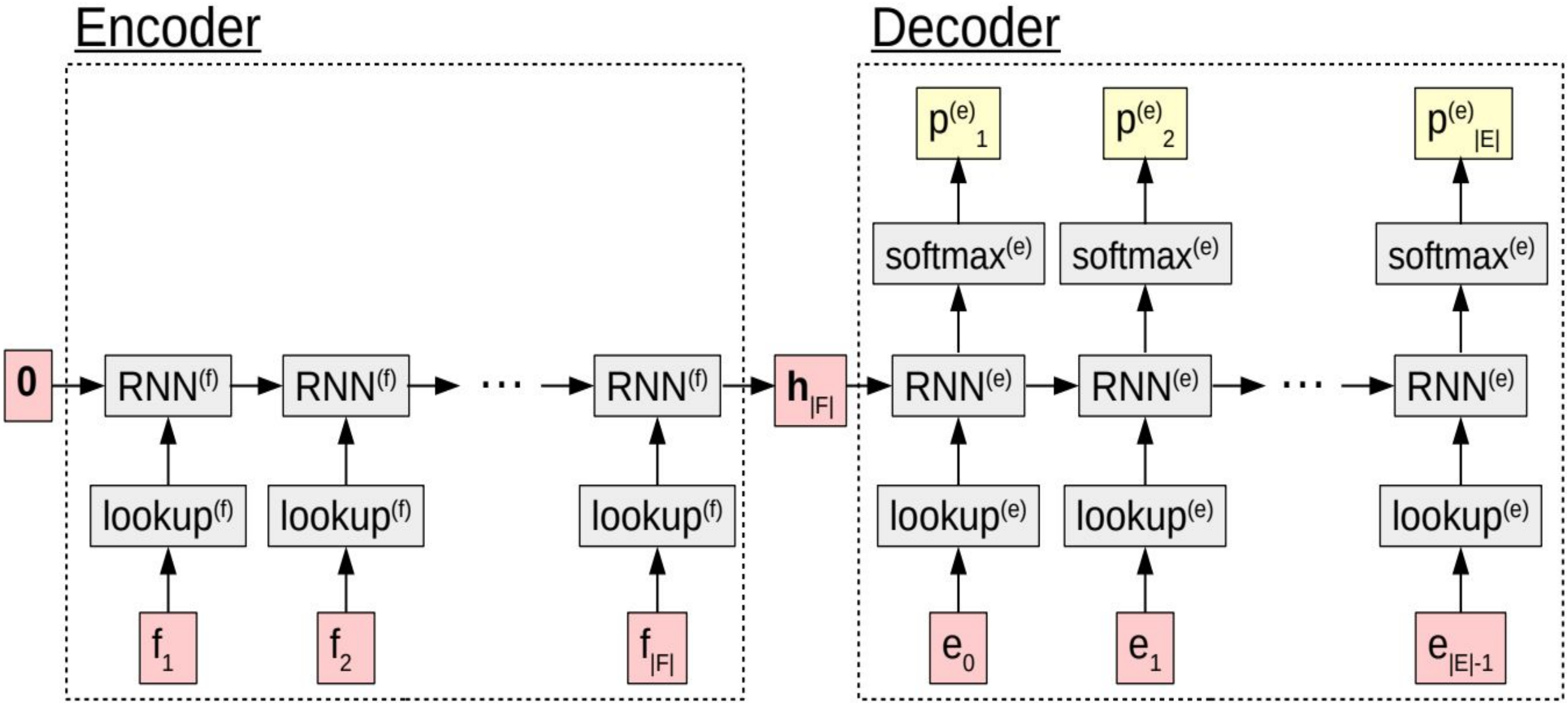


image from https://arxiv.org/pdf/1703.01619v1.pdf

How to utilize all hidden states of the encoder?

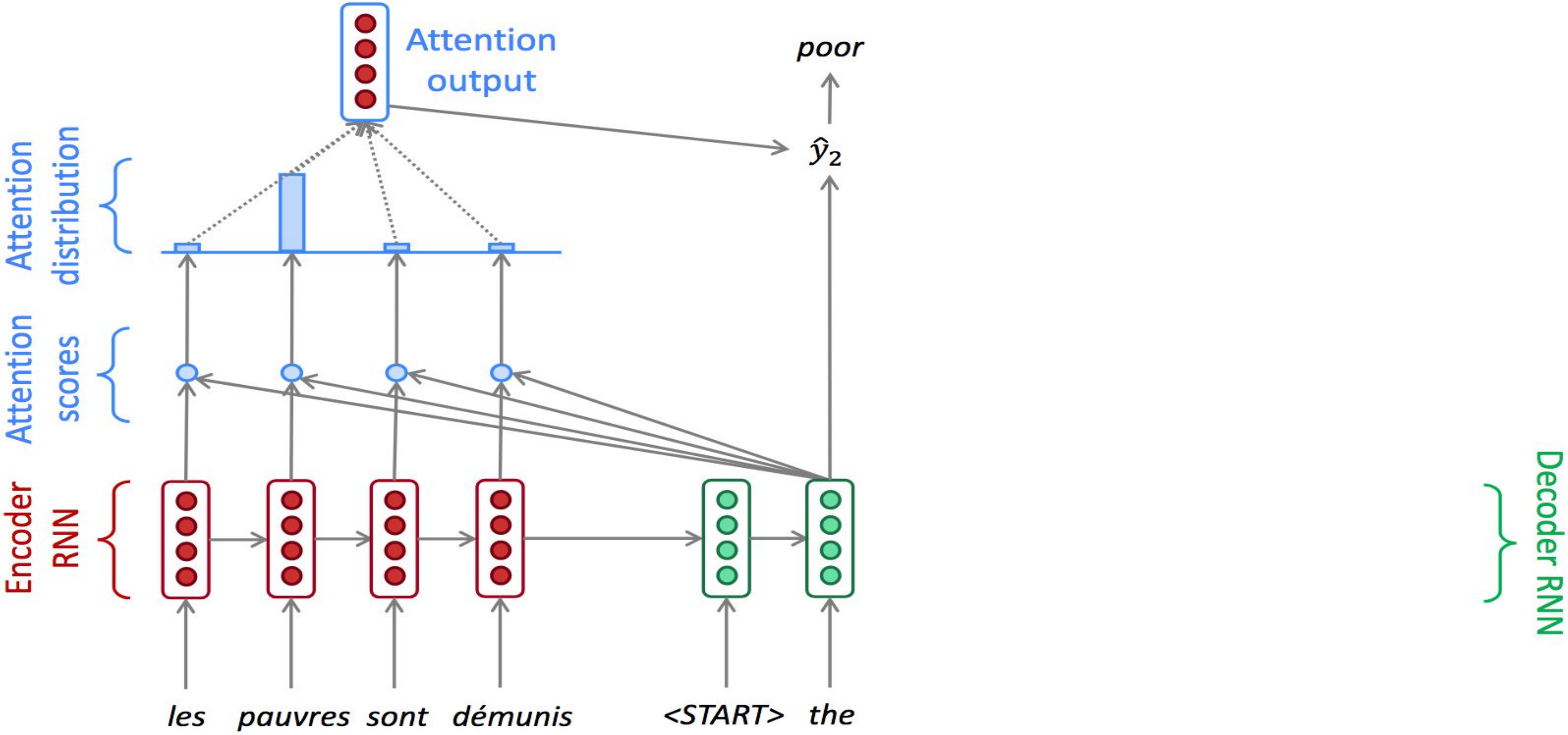
Weighted sum

How to utilize all hidden states of the encoder?

Weighted sum

How to compute weights?

Dot product or a small network

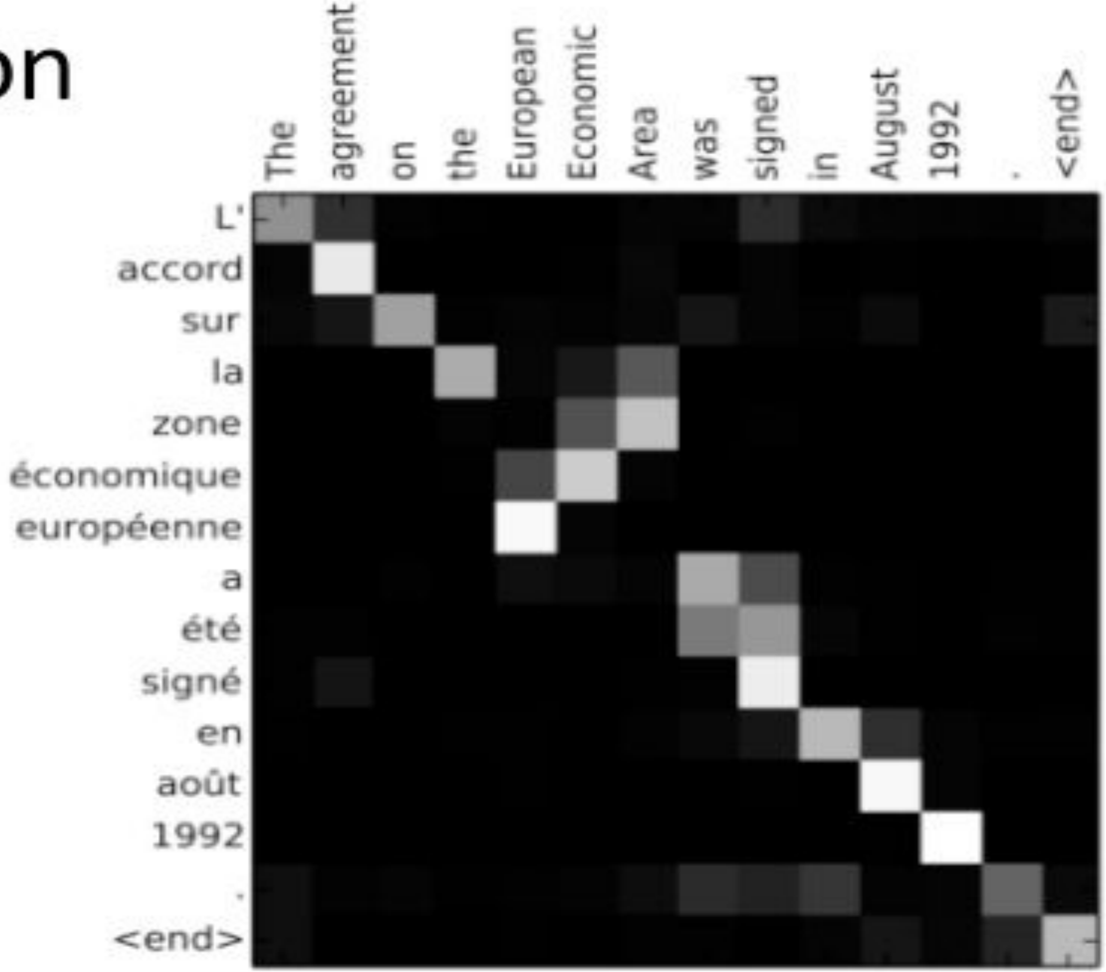


slide from http://web.stanford.edu/class/cs224n/syllabus.html

How to understand seq2seq

1. Select a DL framework (Pytorch is very good)
2. Train a language model
3. Generate text with your LM
4. Train Encoder-Decoder for a simpler task (e.g. POS tagging)
5. Add attention

NMT attention



https://arxiv.org/pdf/1409.0473.pdf

NMT Tips

Data hungry tens of millions sentence pairs

What can be done in low resource situation

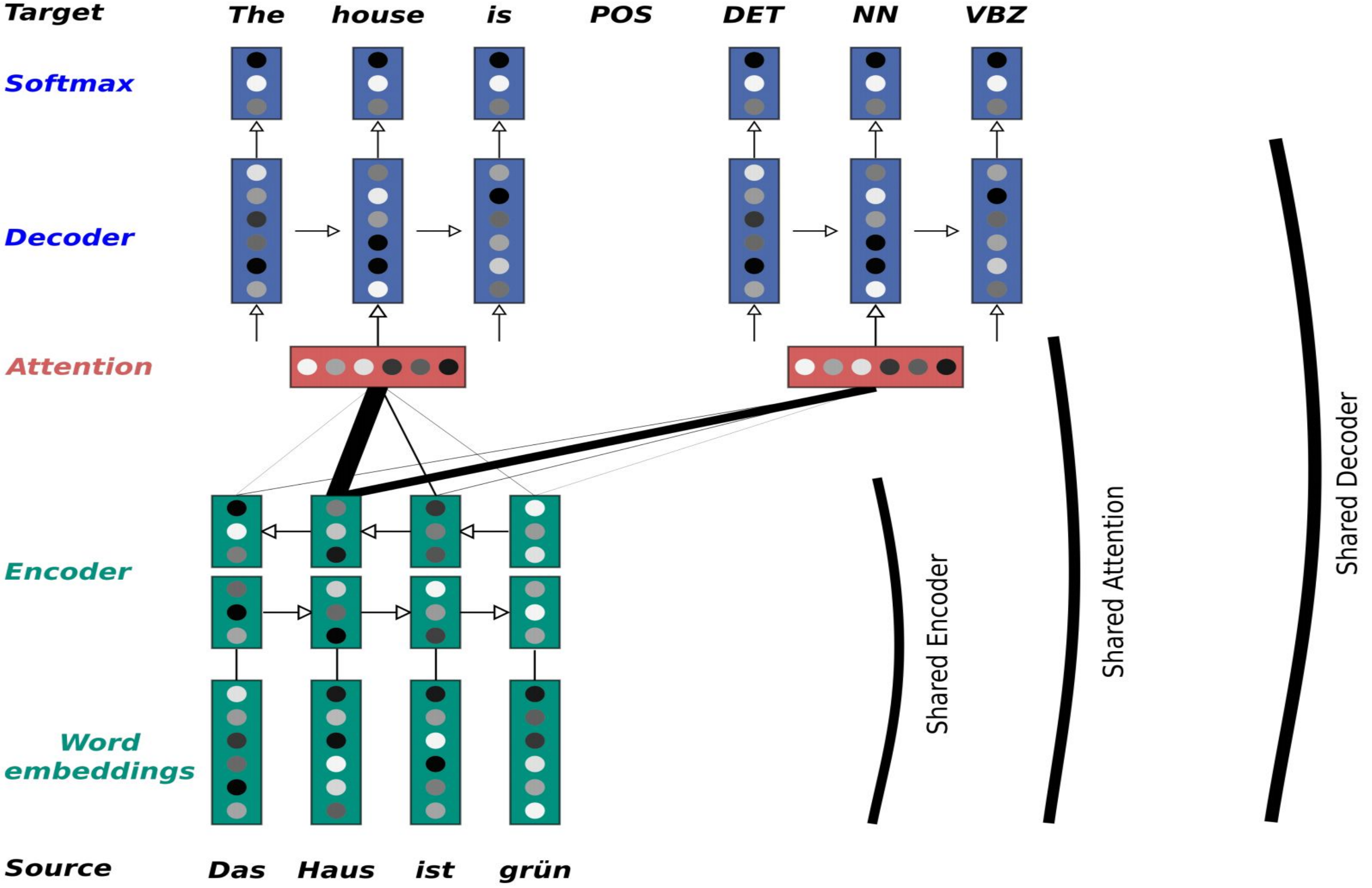
Low resource techniques

Monolingual data

- pretraining
- Backtranslation

Multitask learning

Figure 1: Overview on the different architectures used for multi-task learning



Domain adaptation

Train-Test distribution mismatch

Different domains

Little in-domain data

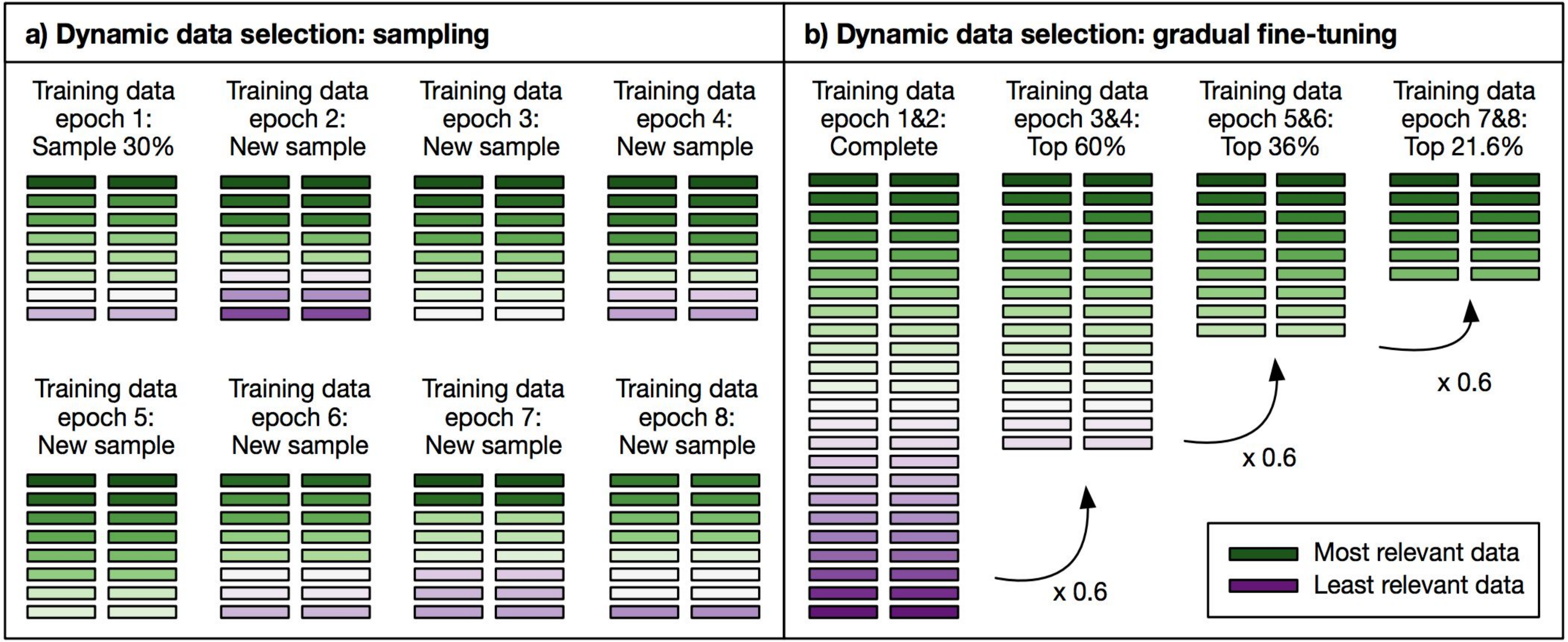


Figure 1: Illustration of two dynamic bitext selection techniques for NMT: *sampling* (left) and *gradual fine-tuning* (right). Measured over 16 training epochs (which is used in this work), the total training time of both examples would be ~30% of the training time needed when using the complete bitext.

Limited vocabulary problem

Subword units (BPE)

Seq2Seq application

- Machine translation
- Speech to text
- Text to speech
- Image captioning
- Visual question answering
- Grammatical error correction
- many, many, others

Questions