DEEP NEURAL NETWORKS FOR NATURAL LANGUAGE PROCESSING

Sergey I. Nikolenko

Steklov Institute of Mathematics at St. Petersburg

PDMI-Huawei Seminar at PDMI RAS
St. Petersburg, October 16, 2017

Random facts:

- October 16 is a big day for female monarchs: in 609, Empress Wu Zetian ascended to the throne of the Tang dynasty, in 1384 Jadwiga of Poland was crowned as King (not a Queen, even though she was a woman), and in 1793 Marie Antoinette was guillotined
- on October 16, 1964, China detonated its first nuclear weapon, and Leonid Brezhnev and Alexei Kosygin were inaugurated as General Secretary and head of government
- in the U.S. and Canada, October 16 is Boss's Day: employees thank their bosses for being kind and fair throughout the year

PLAN

- The deep learning revolution has not left natural language processing alone.
- But in NLP, many problems appear to be “AI-complete”.
- DL in NLP has started with standard architectures (RNN, CNN) but then has branched out into new directions.
- Our plan for today:
  (1) natural language processing: what kind of problems there are;
  (2) what kind of new solutions for these problems stem from deep learning;
  (3) specific example: modern neural architectures for machine translation.

NLP PROBLEMS

NLP PROBLEMS

- Three classes of problems.
- The first class — more “syntactic” problems:
  - they are more or less well-defined,
  - they can usually be posed as classification problems,
  - it is clear how to collect datasets (albeit it may require manual labor, of course).
- These problems were solved reasonably well by classical techniques, but DL improves upon these results.
- But we will see how even “simple” problems require “full-scale understanding” in hard cases.

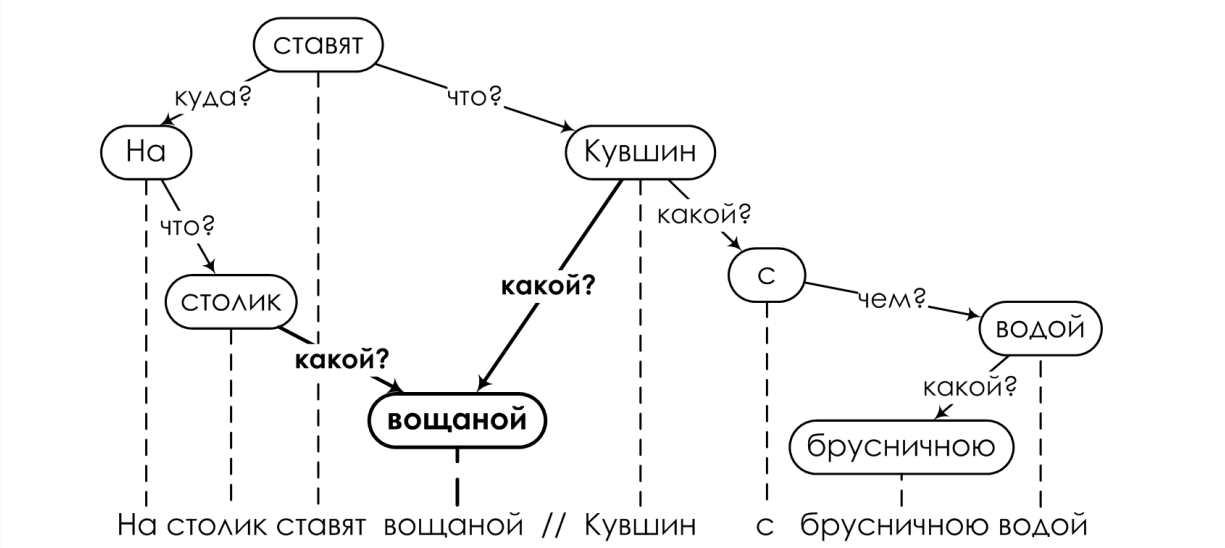
NLP PROBLEMS

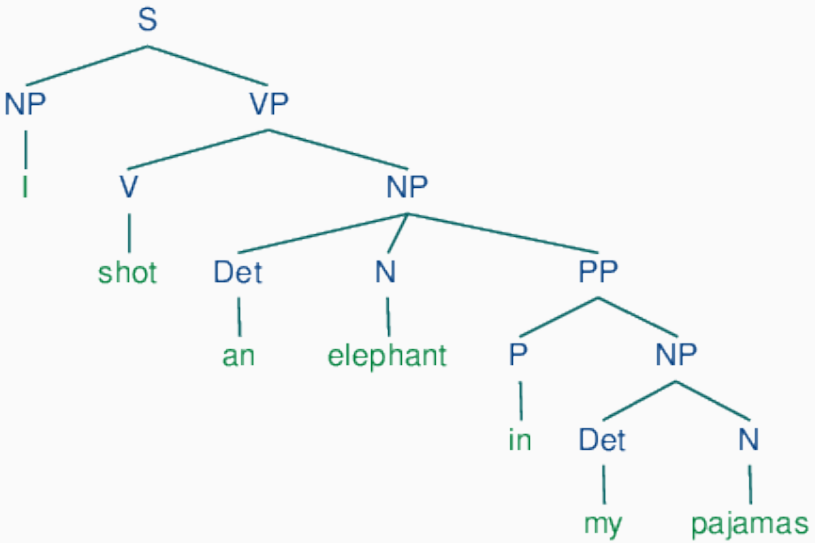
- Part-of-speech tagging:
  - «The panda eats shoots and leaves»
    (ok, this one is about punctuation)
  - «Эти типы стали есть в цехе»
- Morphological segmentation
- Stemming or lemmatization
- Sentence boundary disambiguation:
  - «В 1799 г. А.С. Пушкин родился, в 1812 г. ему исполнилось 13 лет, в 1825 г. — 26 лет, и т. д.»

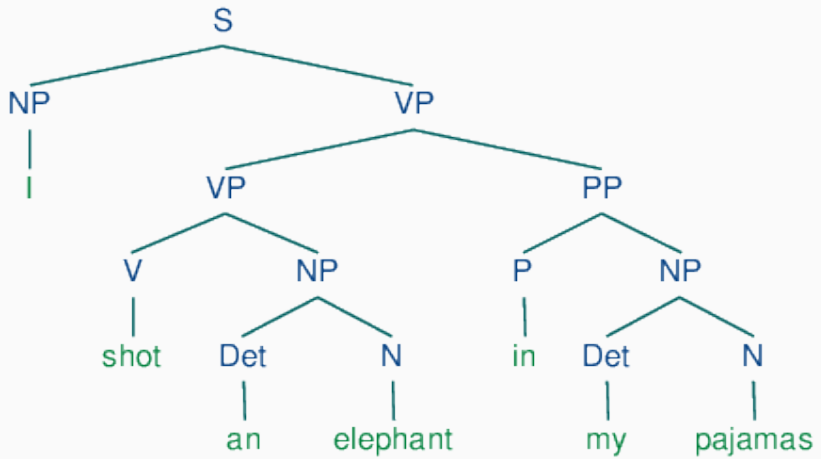
NLP PROBLEMS

- Word segmentation (Asian languages)
- Named entity recognition:
  - «In 2001, Michael Jordan retired from the editorial board of *Machine Learning*»
  - «In 2001, Michael Jordan returned from his second retirement to play for the *Washington Wizards*»
- Word sense disambiguation:
  - «I have a cold today» vs. «We've had a cold day»
  - «After listening to the great bass, Boris Christoff, we ate sea bass at the restaurant»
  - «За песчаной косой лопоухий косой пал под острой косой косой бабы с косой»

- Syntactic parsing:







NLP PROBLEMS

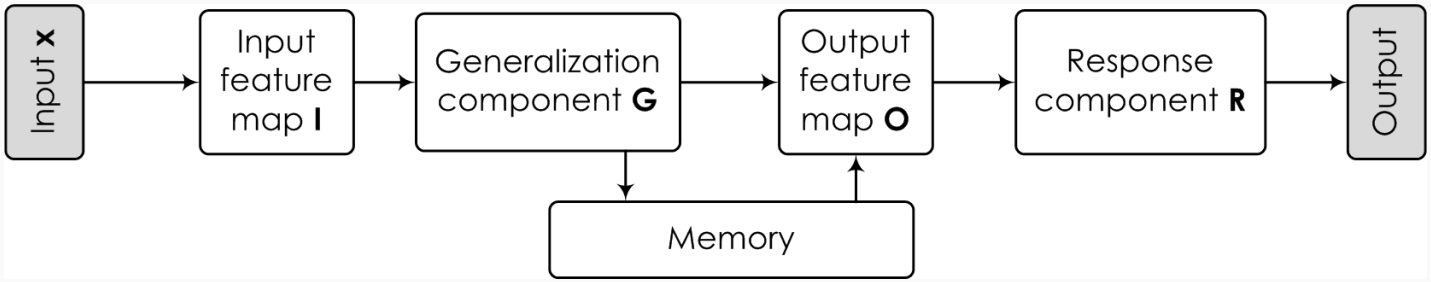
- Coreference resolution, anaphora resolution:
  - «The laptop did not fit in the bag because it was too small»;
  - «The laptop did not fit in the bag because it was too big».
- A similar example in Russian:
  - «Мама вымыла раму, и теперь она блестит»;
  - «Мама вымыла раму, и теперь она устала».
- Pragmatics:
  - «Alice and Betty are mothers»;
  - «Alice and Betty are sisters».
- Big problems with *common sense reasoning*:
  AI models don't have it

NLP PROBLEMS

- Second class – more complex problems that require understanding even more often, but we still know the right answers and can get quality metrics.
- Language modeling:
  - big breakthroughs from RNNs;
  - direct use for speech recognition and the like, but generally the underlying problem for all NLP applications.
- Sentiment analysis:
  - recursive neural networks;
  - requires syntactic parsing first.

NLP PROBLEMS

- Relationship extraction, fact extraction:
  - usually a CNN on vector representations of words + positional embeddings (how far each word is from each entity in the sentence).
- Question answering:
  - formally contains everything else;
  - in reality – only very simple questions:
    - Mary went to the bathroom.
    - John moved to the hallway.
    - Mary travelled to the office.
    - Where is Mary?
  - memory networks, also related to “neural Turing machines”;



  - QA will probably encode “general text understanding”.

NLP PROBLEMS

- Problems where we not only understand text but try to generate new text:
  - text generation per se;
  - automatic summarization;
  - machine translation;
  - dialog and conversational models.
- There are deep learning models for all these problems.

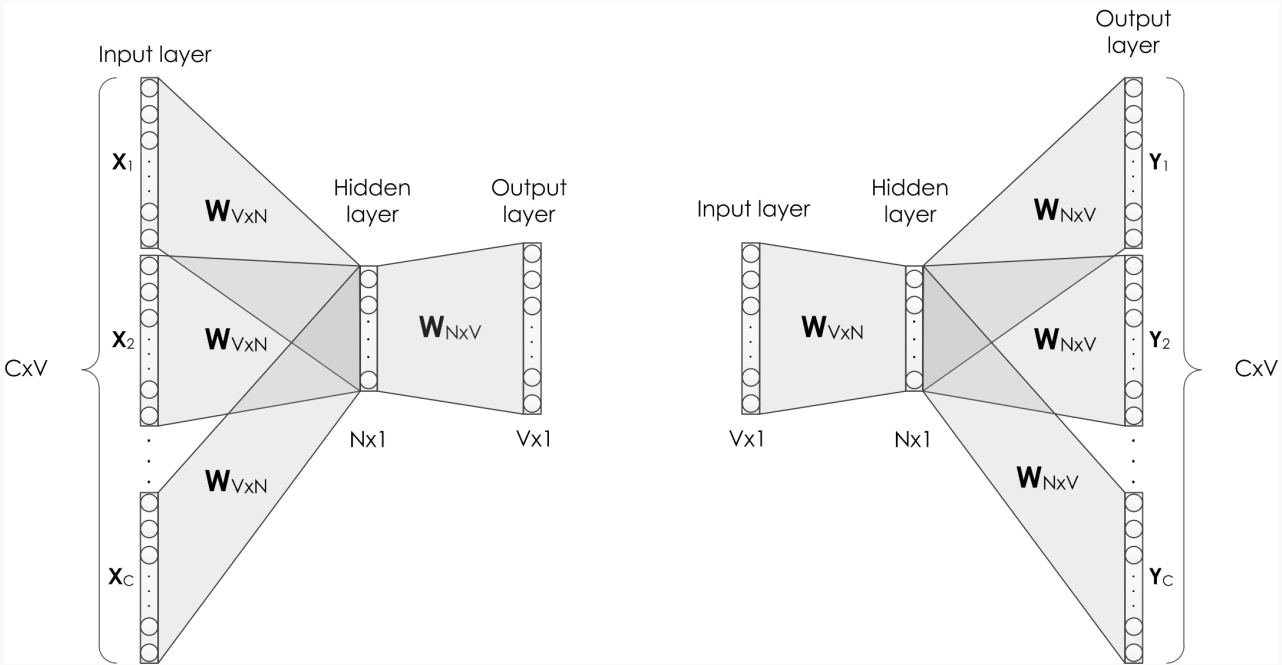
WORD EMBEDDINGS, SENTENCE EMBEDDINGS, AND CHARACTER-LEVEL MODELS

WORD EMBEDDINGS

- Distributional hypothesis in linguistics: words with similar meaning will occur in similar contexts.
- *Distributed word representations* map words to a Euclidean space (usually of dimension several hundred):
  - started in earnest in (Bengio et al. 2003; 2006), although there were earlier ideas;
  - *word2vec* (Mikolov et al. 2013): train weights that serve best for simple prediction tasks between a word and its context: continuous bag-of-words (CBOW) and skip-gram;
  - *Glove* (Pennington et al. 2014): train word weights to decompose the (log) cooccurrence matrix.

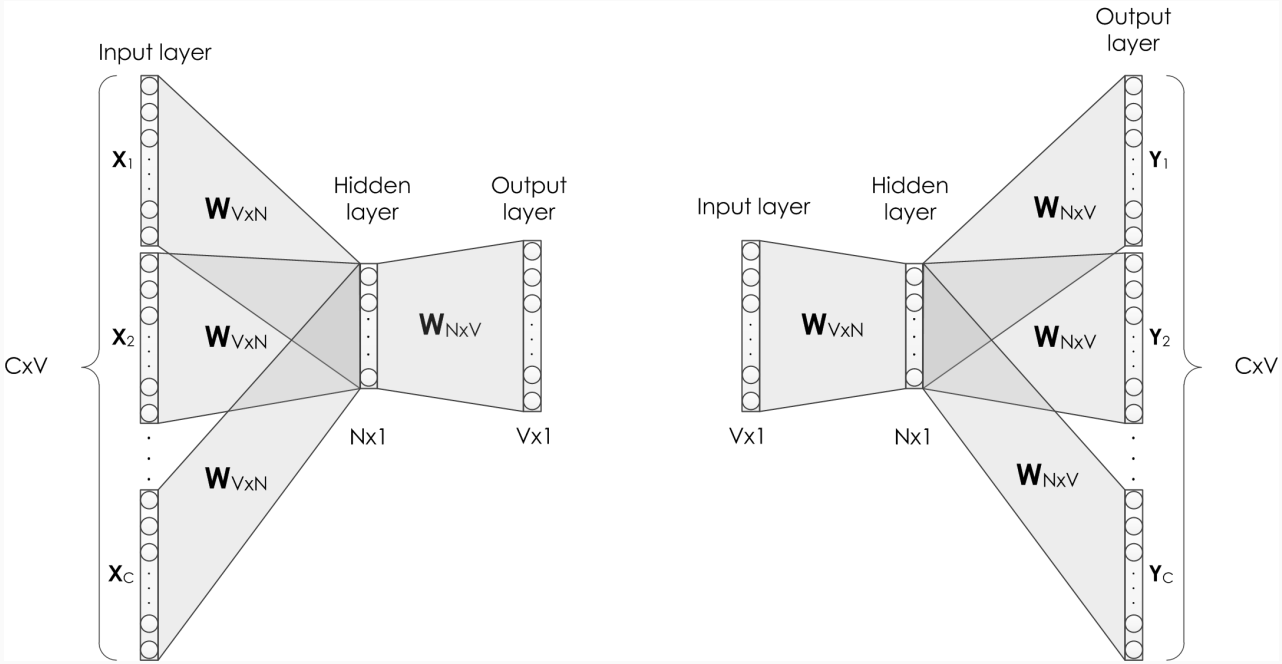
- The CBOW *word2vec* model operates as follows:
  - inputs are one-hot word representations of dimension V;
  - the hidden layer is the matrix of vector embeddings W;
  - the hidden layer's output is the average of input vectors;
  - as output we get an estimate u_j for each word, and

$$\hat{p}(i|c_1, \dots, c_n) = \frac{\exp(u_j)}{\sum_{j'=1}^{V} \exp(u_{j'})}.$$



- In skip-gram, it's the opposite:
  - we predict each context word from the central word;
  - so now there are several multinomial distributions, one $\mathrm{softmax}$ for each context word:

$$\hat{p}(c_k|i) = \frac{\exp(u_{kc_k})}{\sum_{j'=1}^{V} \exp(u_{j'})}.$$



- GloVe – we are trying to approximate the cooccurrence matrix $X \in \mathbb{R}^{V \times V}$:

$$p_{ij} = p(j \mid i) = \frac{X_{ij}}{X_i} = \frac{X_{ij}}{\sum_k X_{ik}}.$$

- More precisely, the ratios $\frac{p_{ij}}{p_{kj}}$.
- Example from the Russian wiki:

Word k	No. of occurrences			Probabilities $p(k \mid ...)$, $\times 10^{-4}$		Ratio $\frac{p(k\mid\text{клуб})}{p(k\mid\text{команда})}$
	Total	Together with:				
		клуб	команда	клуб	команда	
футбол	29988	54	34	18.0	11.3	1.588
хоккей	10957	16	7	6.39	14.6	2.286
гольф	2721	11	1	40.4	3.68	11.0
корабль	100127	0	30	0.0	3.00	0.0

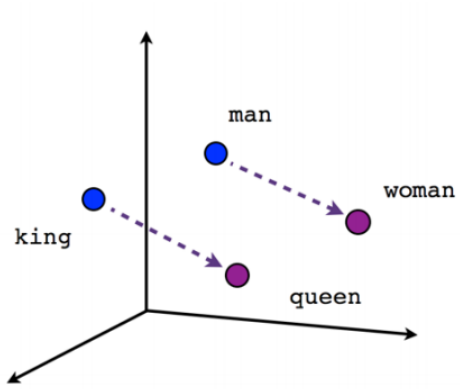
- In any case, we get a vector representation for every word.

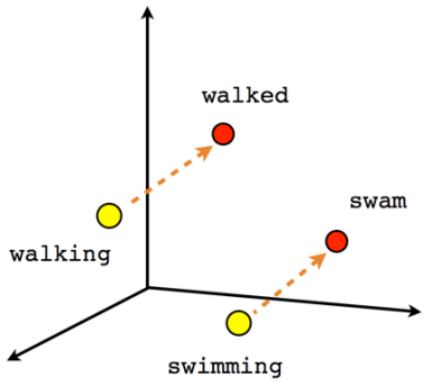
WORD EMBEDDINGS

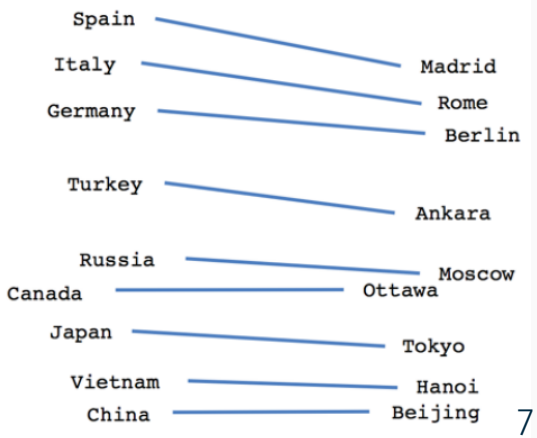
- As a result, close vectors are semantically similar:

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

- Plus linear relations between concepts:

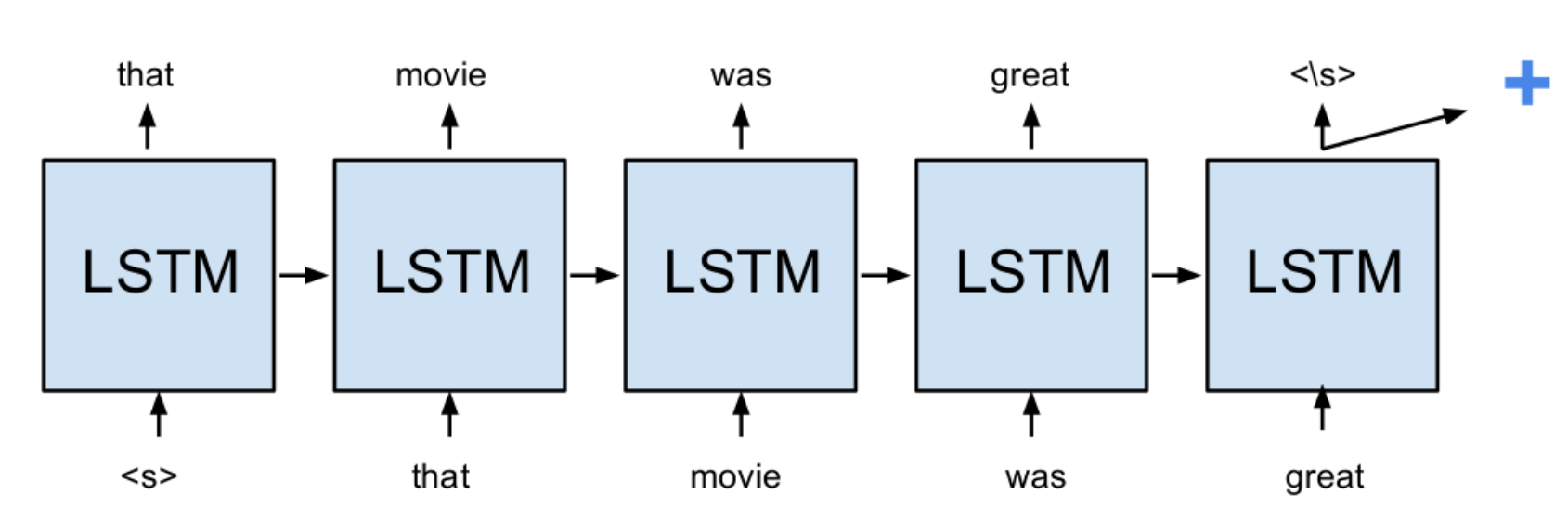






HOW TO USE WORD VECTORS

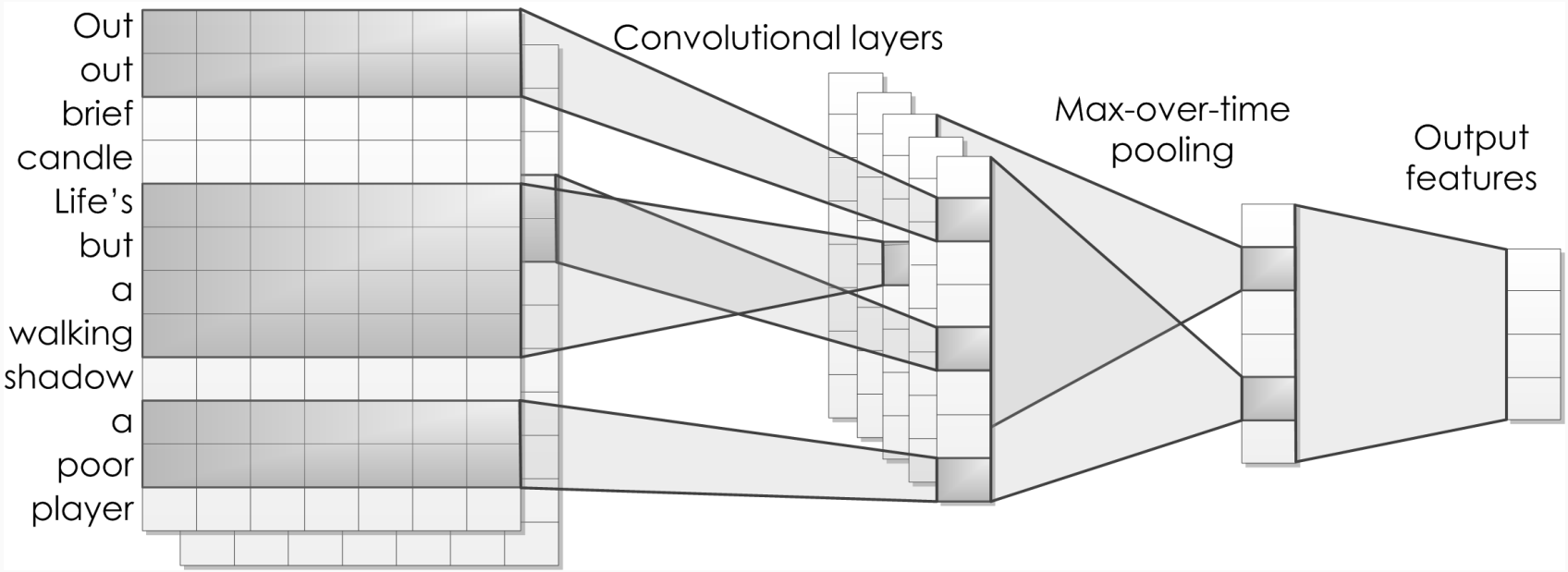
- Next we can use recurrent architectures on top of word vectors.
- E.g., LSTMs for sentiment analysis:



- Train a network of LSTMs for language modeling, then use either the last output or averaged hidden states for sentiment.

HOW TO USE WORD VECTORS

- Or a CNN with one-dimensional convolutions:

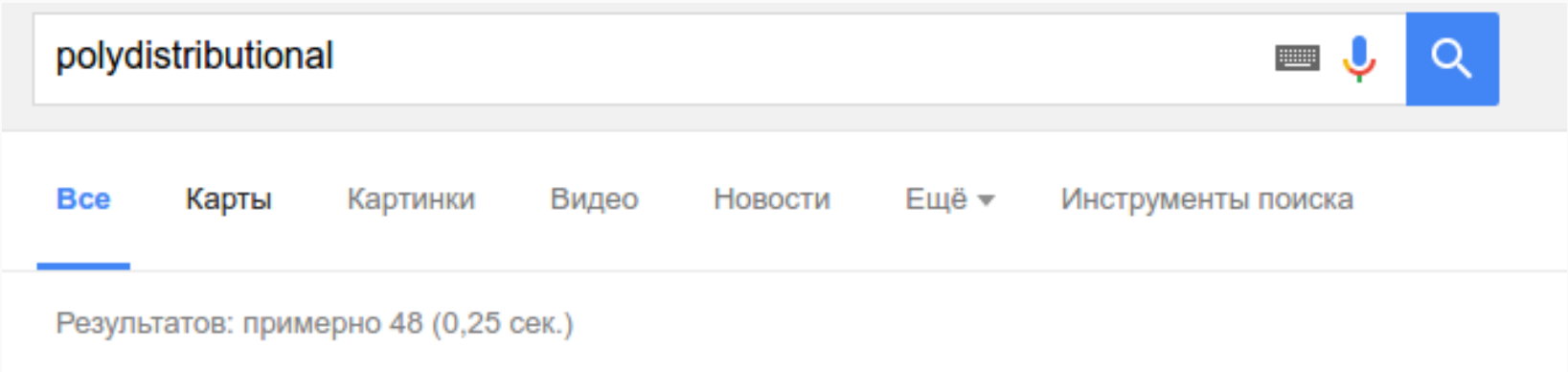


Up and down from word embeddings

- Word embeddings are the first step of most DL models in NLP.
- But we can go both up and down from word embeddings.
- First, a sentence is not necessarily the sum of its words:
  - but to a first approximation, it is;
  - there are various constructions of “paragraph vectors”;
  - and even more encoder-like architectures.
- Second, a word is not quite as atomic as the word2vec model would like to think.

CHARACTER-LEVEL MODELS

- Word embeddings have important shortcomings:
  - vectors are independent but words are not; consider, in particular, morphology-rich languages like Russian;
  - the same applies to out-of-vocabulary words: a word embedding cannot be extended to new words;
  - word embedding models may grow large; it's just lookup, but the whole vocabulary has to be stored in memory with fast access.
- E.g., "polydistributional" gets 48 results on Google, so you probably have never seen it, and there's very little training data:



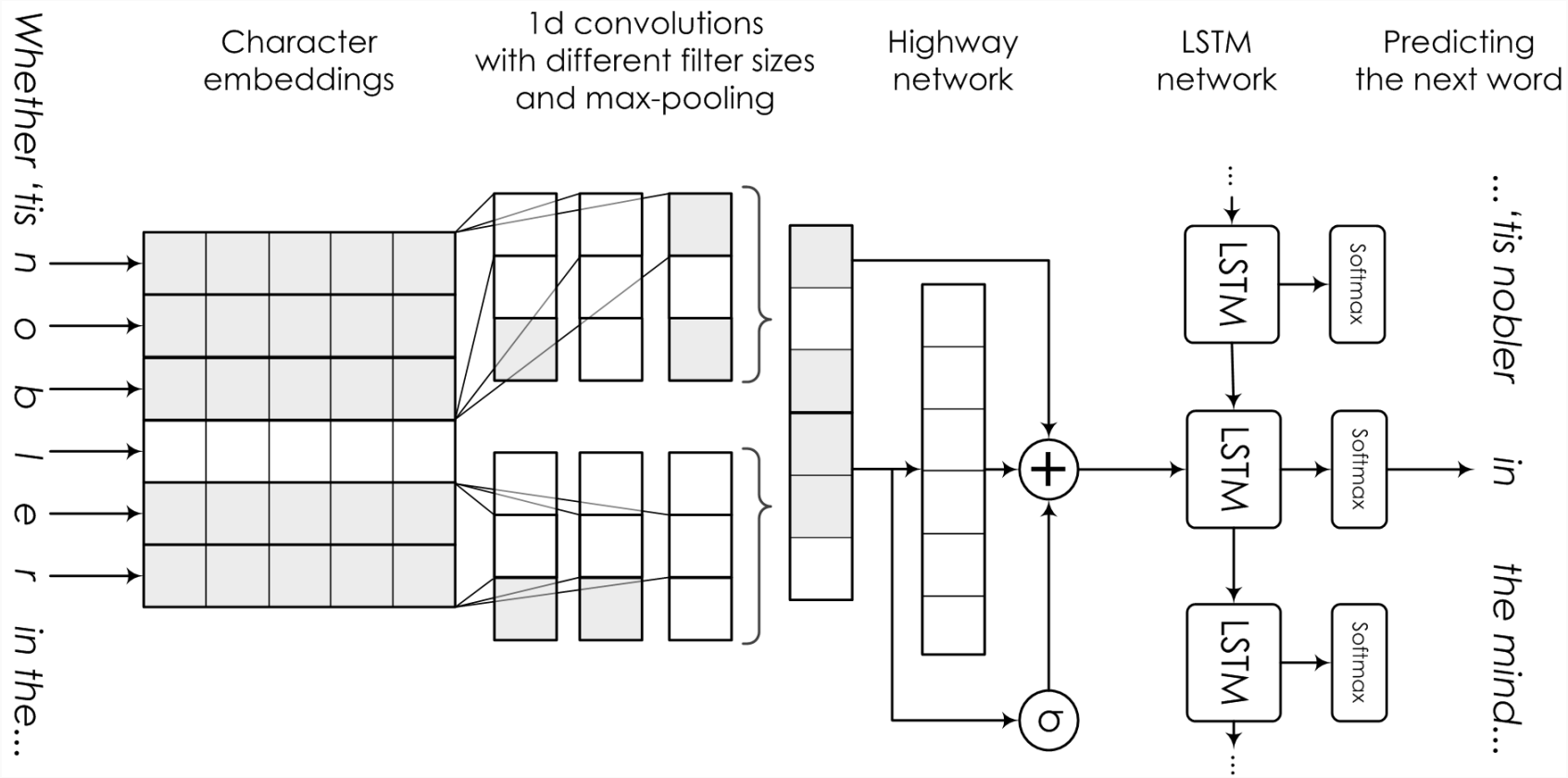
- Do you have an idea what it means? Me too.

Character-level models

- Hence, *character-level representations*:
  - began by decomposing a word into morphemes (Luong et al. 2013; Botha and Blunsom 2014; Soricut and Och 2015);
  - but this adds errors since morphological analyzers are also imperfect, and basically a part of the problem simply shifts to training a morphology model;
  - two natural approaches on character level: LSTMs and CNNs;
  - in any case, the model is slow but we do not have to apply it to every word, we can store embeddings of common words in a lookup table as before and only run the model for rare words – a nice natural tradeoff.

Modern char-based language model: Kim et al., 2015

- Sample modern character-based language model (Kim et al., 2015):



- Unites CNN, RNN, highway networks, embeddings...

MACHINE TRANSLATION

Evaluation for sequence-to-sequence models

- Next we will consider specific models for machine translation.
- But how do we evaluate NLP models that produce text?
- Quality metrics for comparing with reference sentences produced by humans:
  - BLEU (Bilingual Evaluation Understudy): reweighted precision (incl. multiple reference translations);
  - METEOR: harmonic mean of unigram precision and unigram recall;
  - TER (Translation Edit Rate): number of edits between the output and reference divided by the average number of reference words;
  - LEPOR: combine basic factors and language metrics with tunable parameters.
- The same metrics apply to paraphrasing and, generally, all problems where the (supervised) answer should be a free-form text.
- There is one problem...

EVALUATION FOR SEQUENCE-TO-SEQUENCE MODELS

- They don't work at all!

	Twitter				Ubuntu			
Metric	**Spearman**	p-value	**Pearson**	p-value	**Spearman**	p-value	**Pearson**	p-value
Greedy	0.2119	0.034	0.1994	0.047	0.05276	0.6	0.02049	0.84
Average	0.2259	0.024	0.1971	0.049	-0.1387	0.17	-0.1631	0.10
Extrema	0.2103	0.036	0.1842	0.067	0.09243	0.36	-0.002903	0.98
METEOR	0.1887	0.06	0.1927	0.055	0.06314	0.53	0.1419	0.16
BLEU-1	0.1665	0.098	0.1288	0.2	-0.02552	0.8	0.01929	0.85
BLEU-2	0.3576	< 0.01	0.3874	< 0.01	0.03819	0.71	0.0586	0.56
BLEU-3	0.3423	< 0.01	0.1443	0.15	0.0878	0.38	0.1116	0.27
BLEU-4	0.3417	< 0.01	0.1392	0.17	0.1218	0.23	0.1132	0.26
ROUGE	0.1235	0.22	0.09714	0.34	0.05405	0.5933	0.06401	0.53
Human	0.9476	< 0.01	1.0	0.0	0.9550	< 0.01	1.0	0.0

Table 3: Correlation between each metric and human judgements for each response. Correlations shown in the human row result from randomly dividing human judges into two groups.

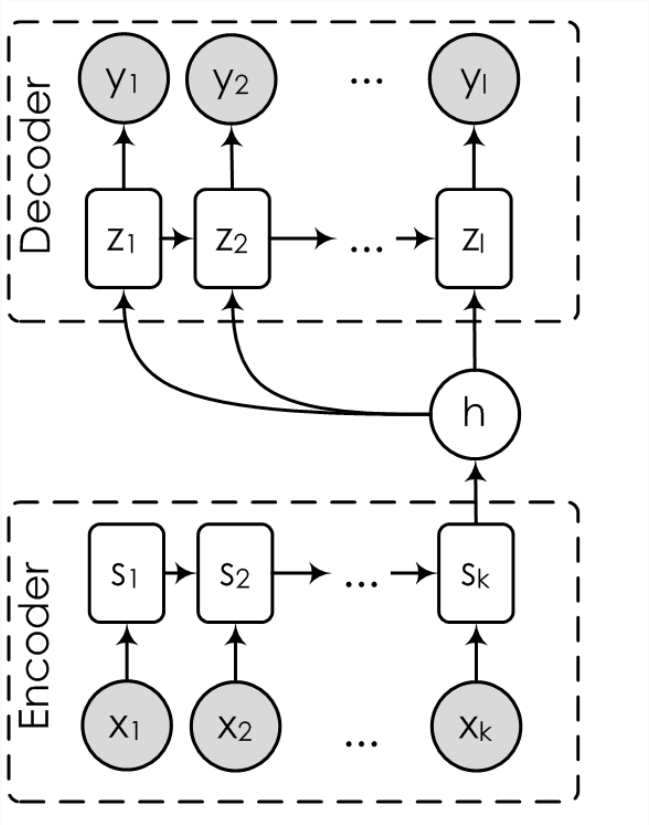
- Well, actually they do work, but it's more complicated than it seems.

MACHINE TRANSLATION

- Translation is a very convenient problem for modern NLP:
  - on one hand, it is very practical, obviously important;
  - on the other hand, it's very high-level, virtually impossible without deep understanding, so if we do well on translation, we probably do something right about understanding;
  - on the third hand (oops), it's quantifiable (BLEU, TER etc.) and has relatively large available datasets (parallel corpora).

ENCODER-DECODER ARCHITECTURES

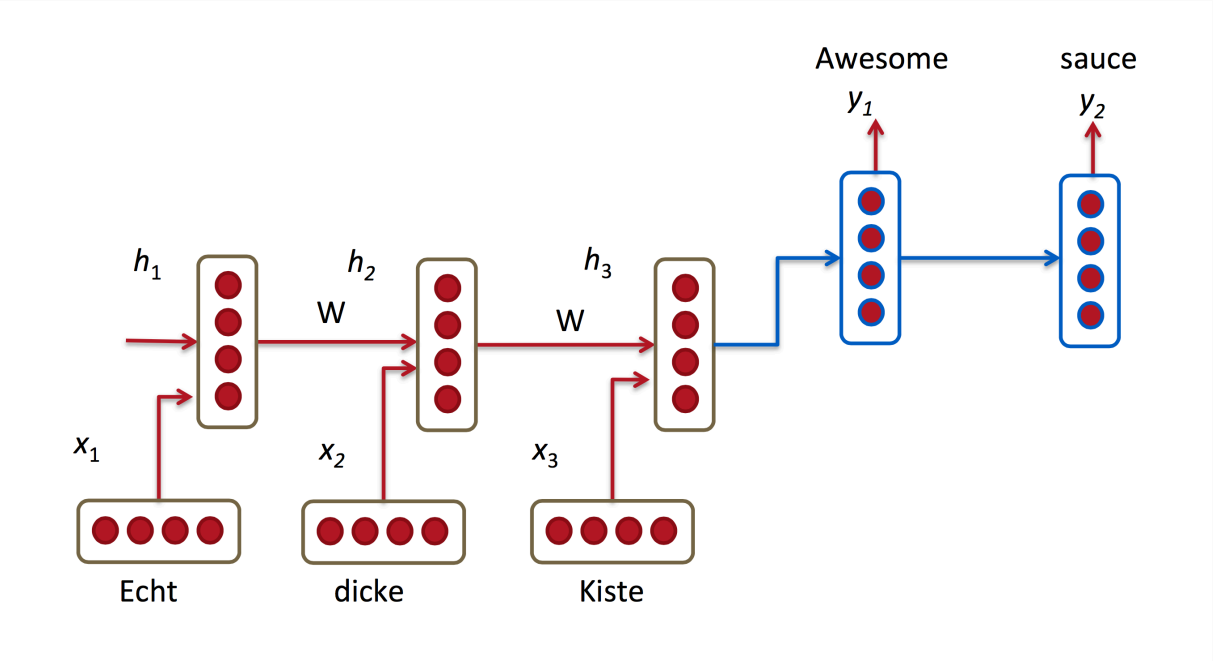
- Encoder-decoder architectures (Sutskever et al., 2014; Cho et al., 2014):



- First code, then decode back.

ENCODER-DECODER ARCHITECTURES

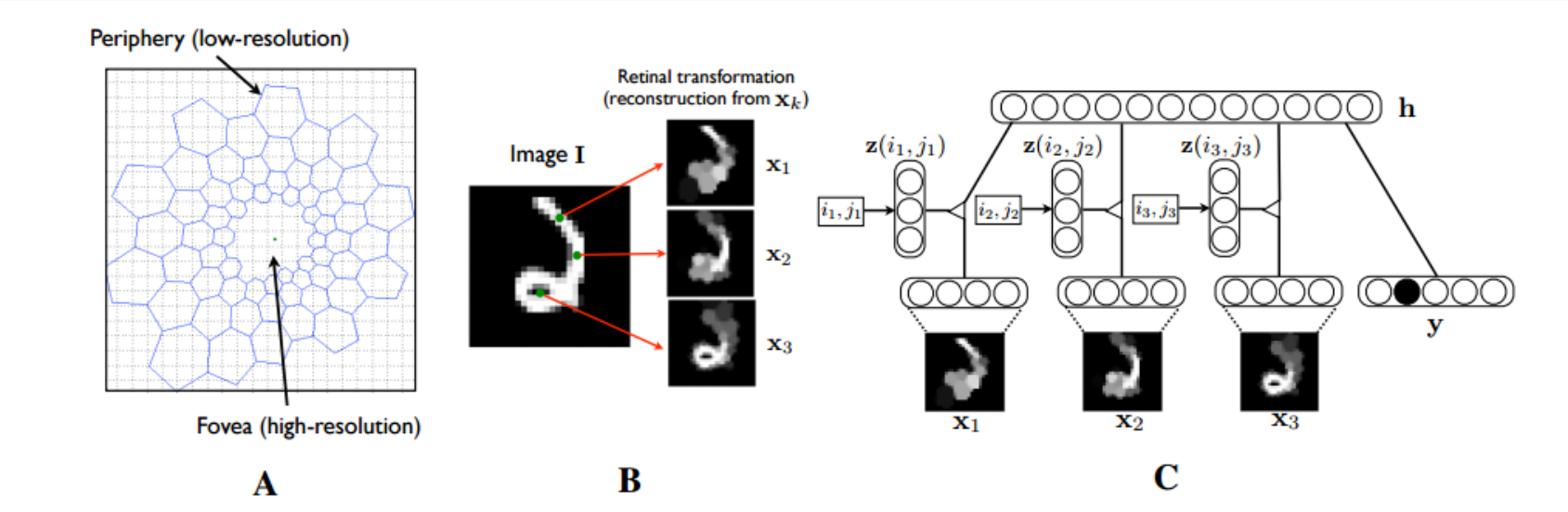
- The same idea works with translation.



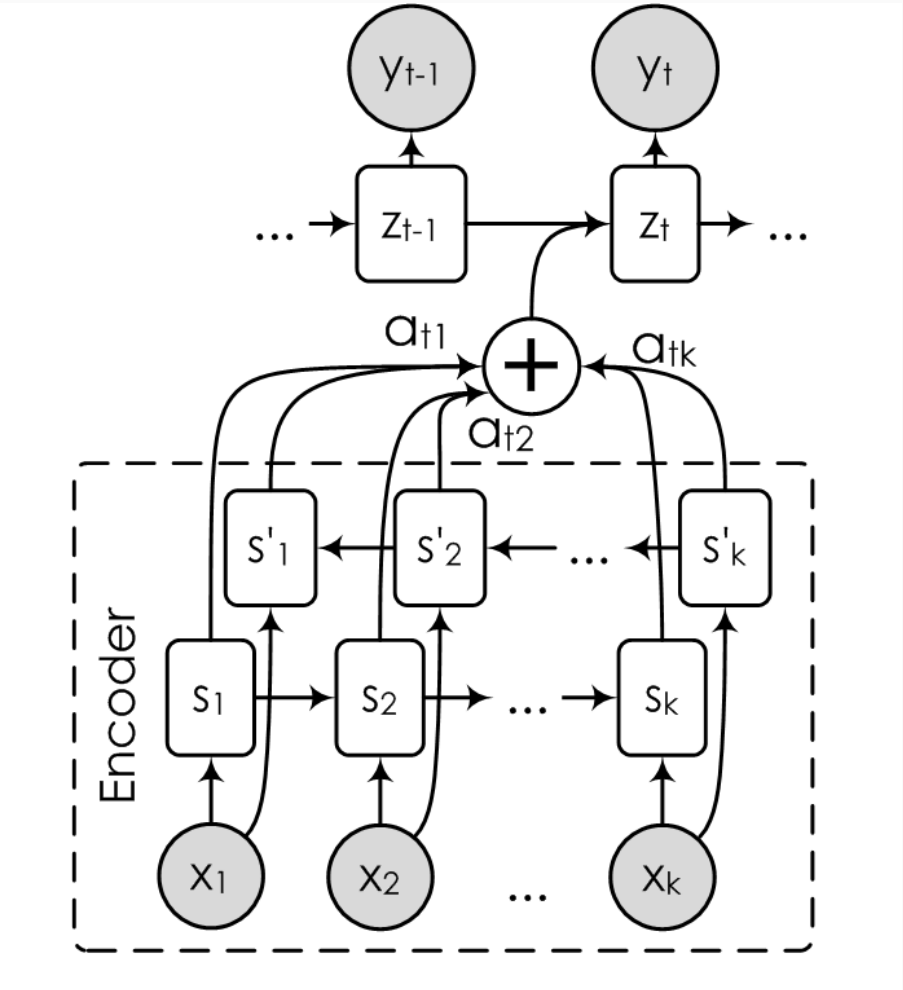
- Problem: we need to compress the entire sentence into a single vector.
- And it does not work at all with longer fragments...

ATTENTION IN NEURAL NETWORKS

- Possible solution: train special weights that show how important a certain part of the input is for the currently generated part of the output.
- This is somewhat similar to human *attention*: what do we put into working memory?
- First applications in NNs – foveal glimpses with RBMs (Larochelle, Hinton, 2010)

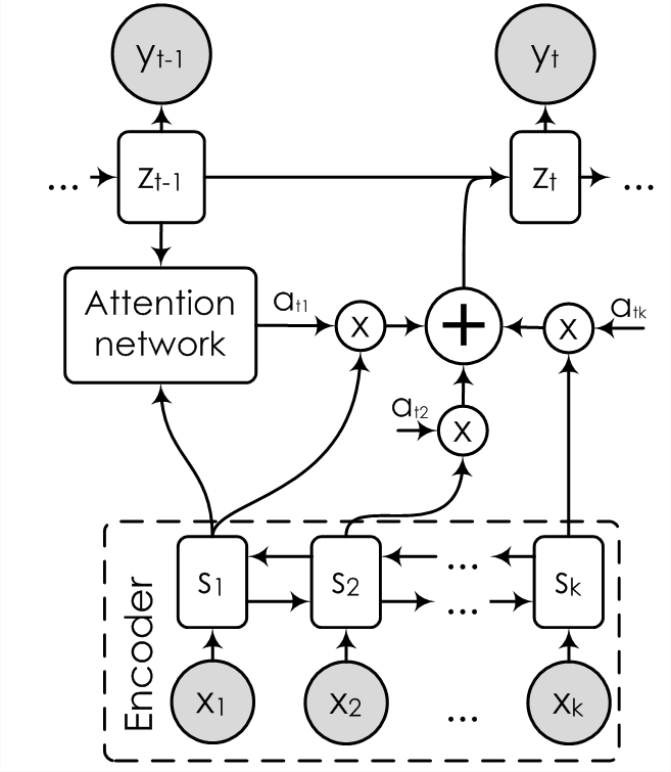


- A direct application – bidirectional LSTM + attention (Bahdanau et al. 2014):



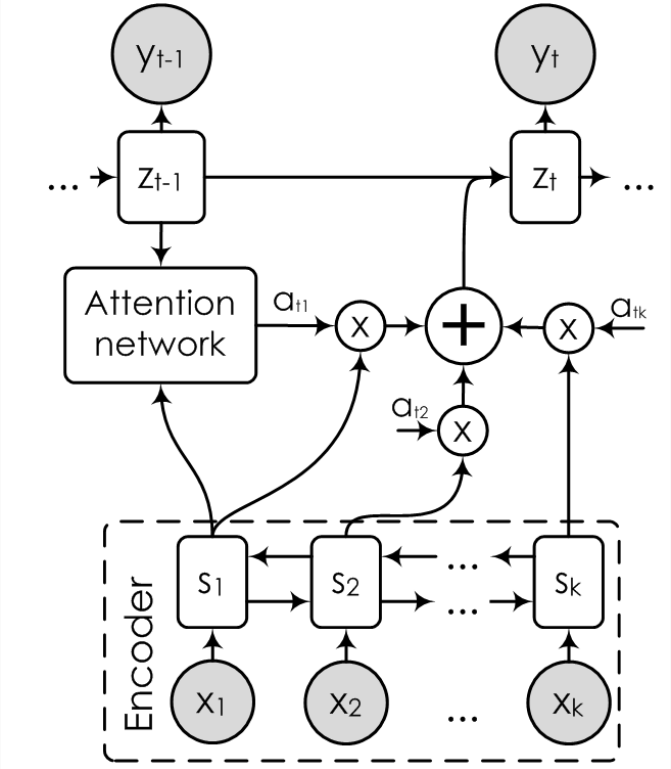
ATTENTION IN NEURAL NETWORKS

- *Soft attention* (Luong et al. 2015a; 2015b; Jean et al. 2015):
  - encoder is a bidirectional RNN;
  - attention network estimates relevance: are we translating this word right now?



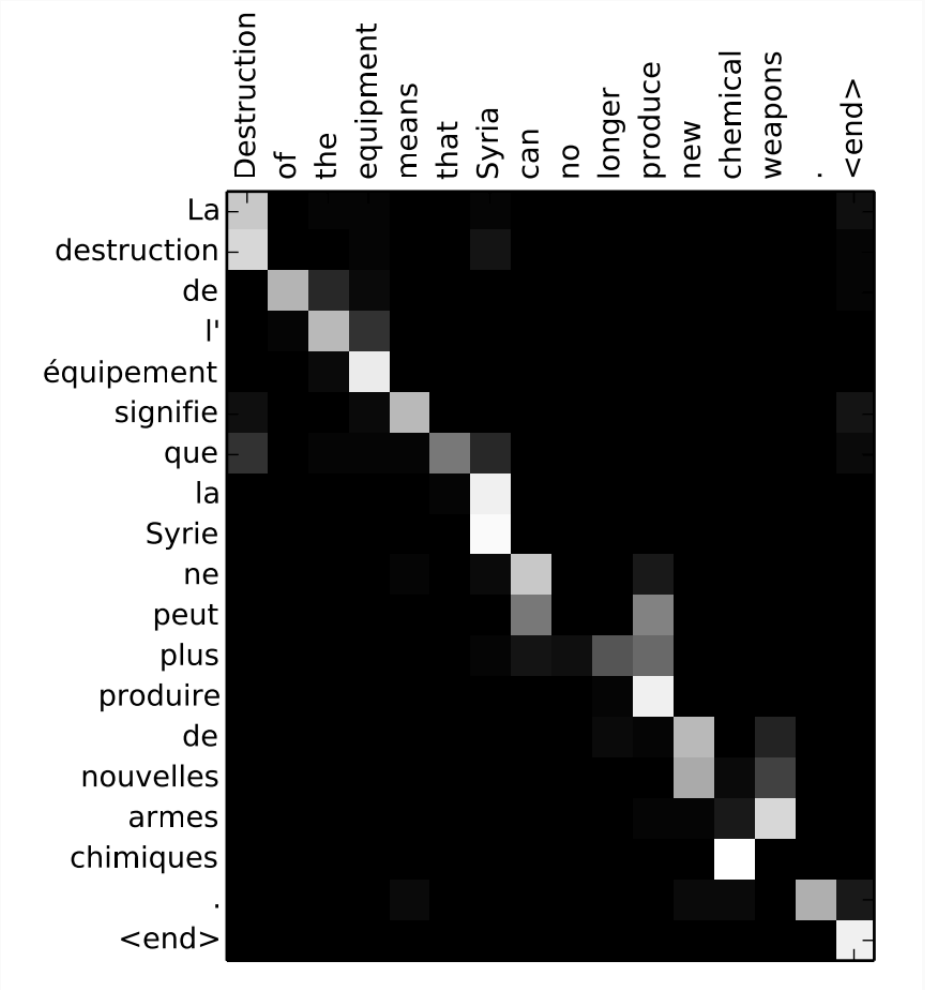
- Formally very simple: compute attention weights α_{tj} and re-weigh context vectors:

$$e_{tj} = a(z_{t-1}, j), \quad \alpha_{tj} = \mathrm{softmax}(e_{tj}; e_{t*}),$$
$$c_t = \sum_j \alpha_{tj} h_j, \text{ and now } z_t = f(s_{t-1}, y_{t-1}, c_i).$$

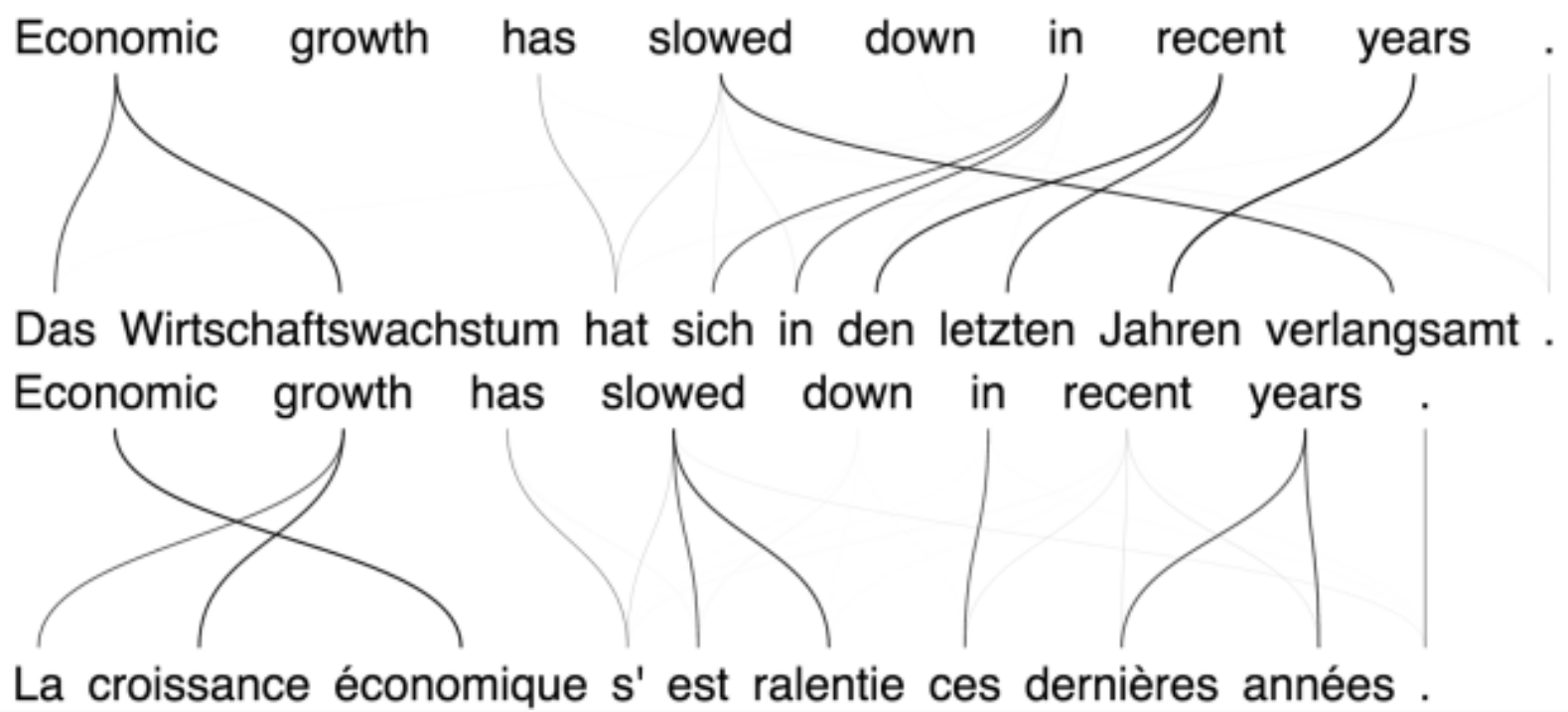


ATTENTION IN NEURAL NETWORKS

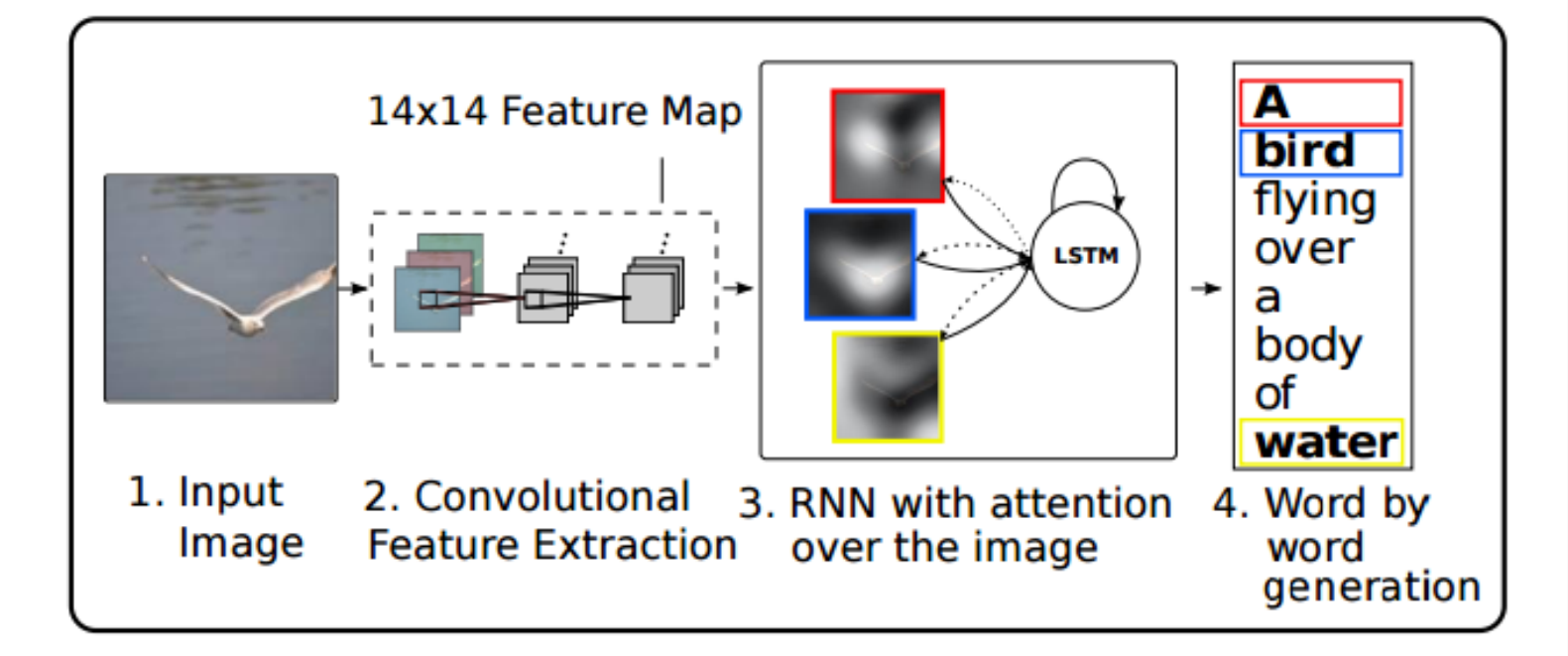
- As a result we can visualize what the network is looking at:



- The word order is much better this way:

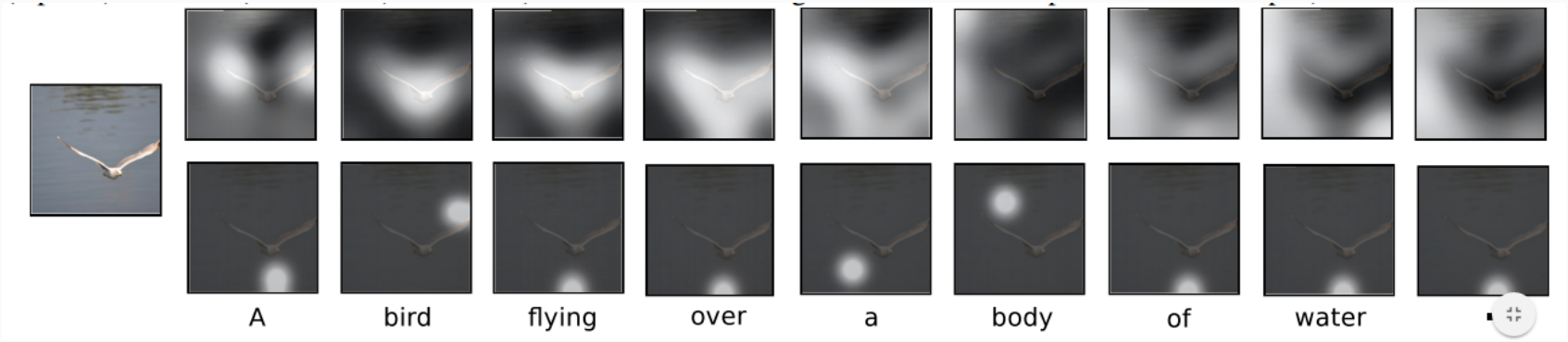


- Other applications of attention mechanisms can be NLP-related too.
- Show, Attend, and Tell (Xu et al., 2015): descriptions of images.



ATTENTION IN NEURAL NETWORKS

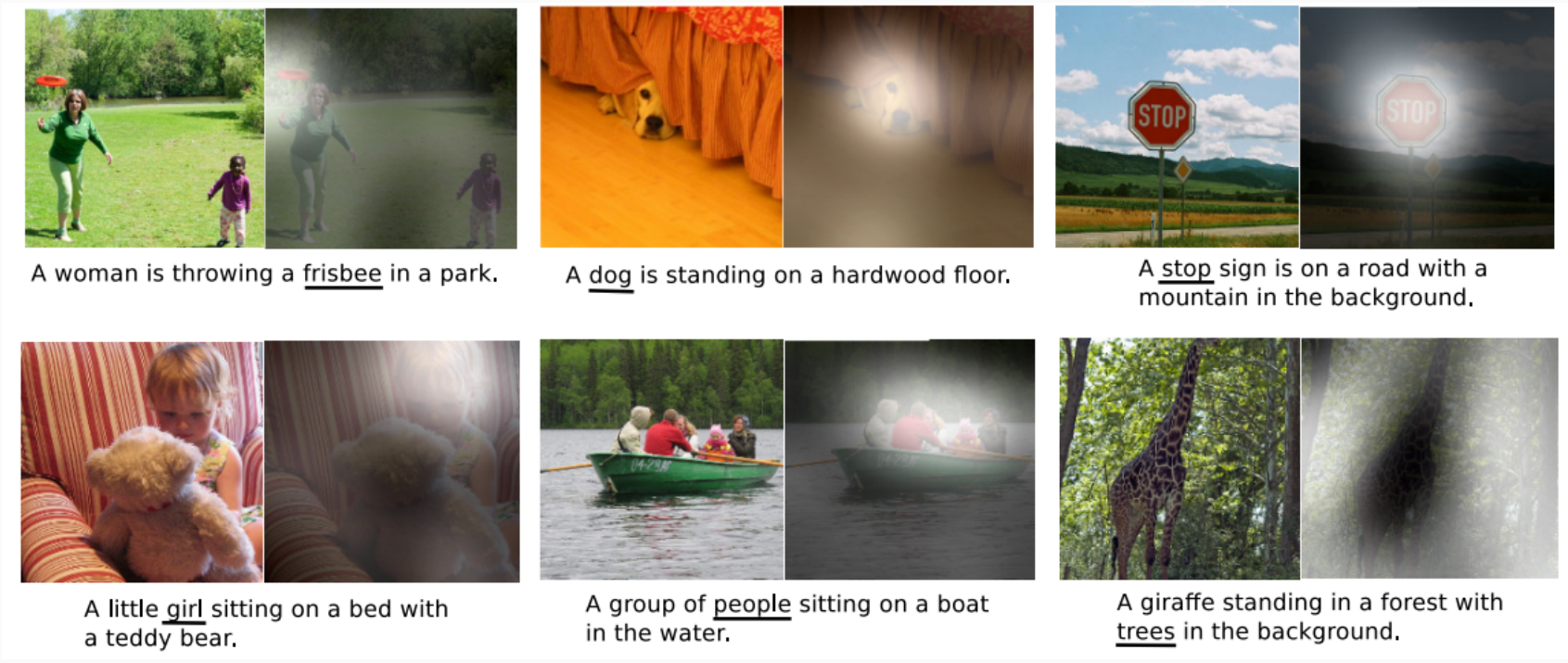
- Soft attention vs. hard attention (stochastically choose a specific part of the image):



- Hard attention is trained by maximizing a variational lower bound.

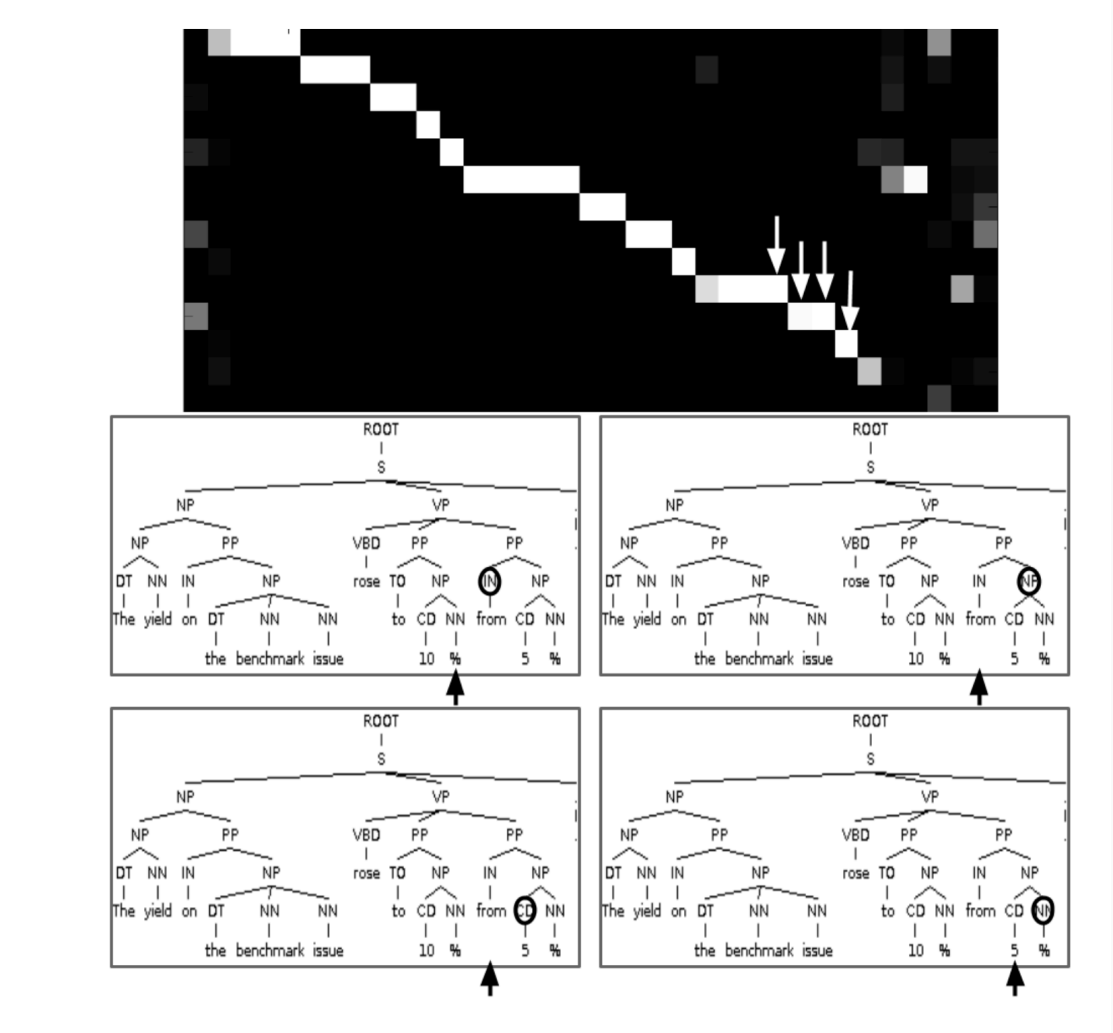
ATTENTION IN NEURAL NETWORKS

- Often pretty good results:



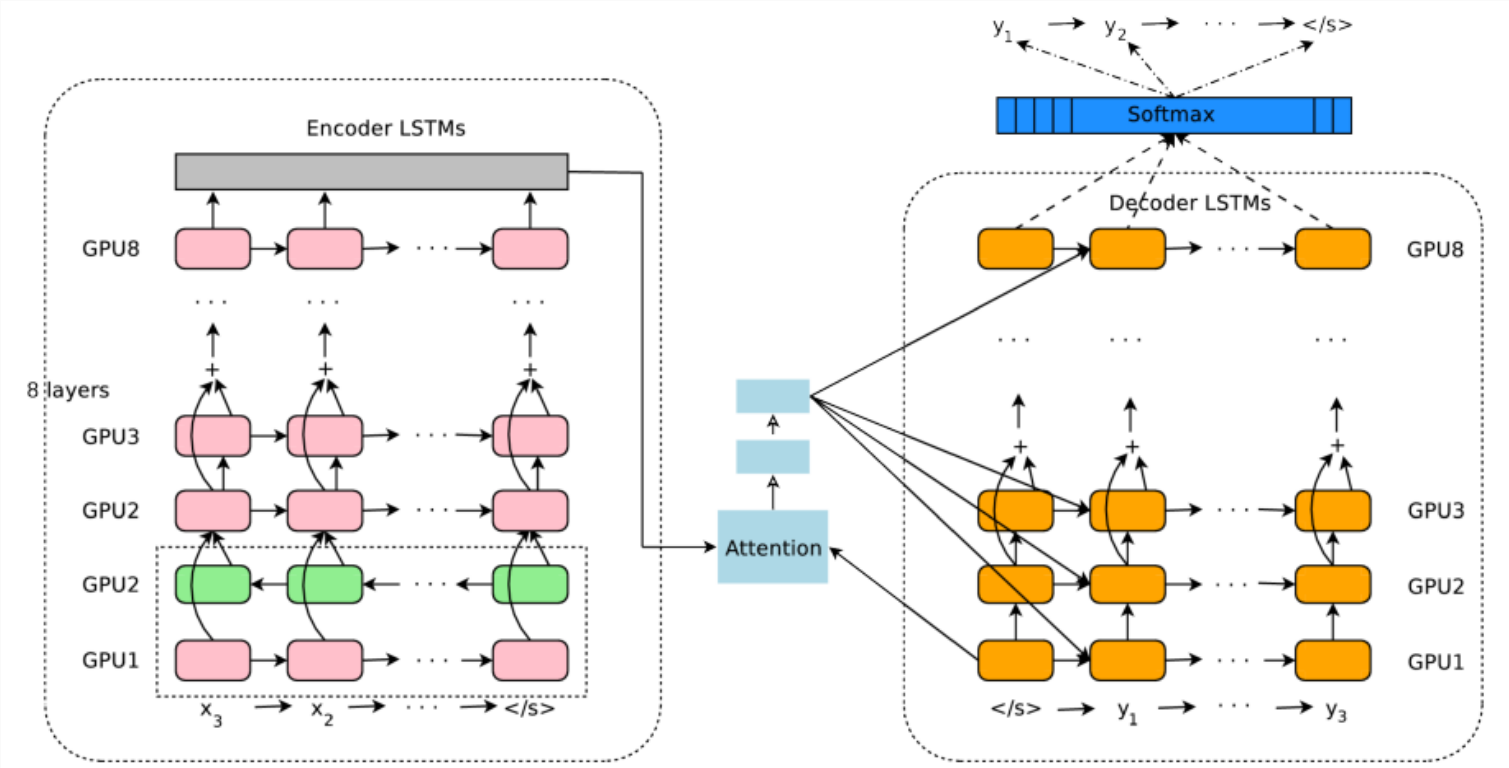
ATTENTION IN NEURAL NETWORKS

- Even closer – «Grammar as a Foreign Language» (Vinyals et al., 2015)



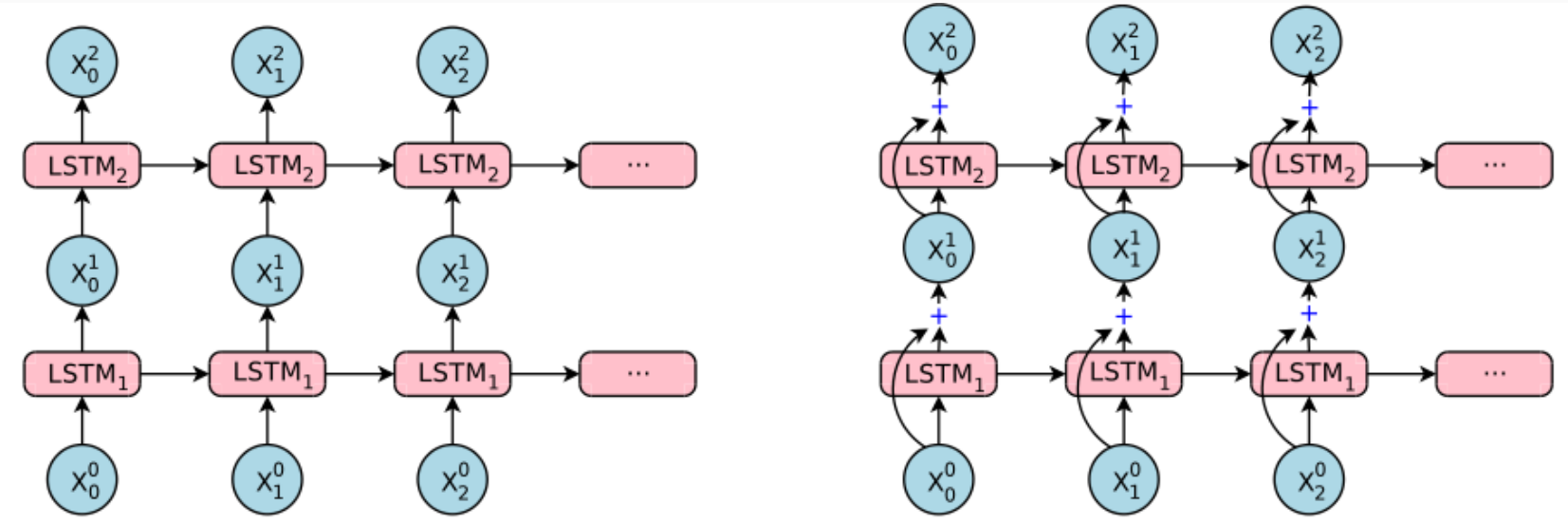
GOOGLE TRANSLATE

- September 2016: Wu et al., *Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation*:
  - shows how Google Translate actually works;
  - the basic architecture is the same: encoder, decoder, attention;
  - RNNs have to be deep enough to capture language irregularities, so 8 layers for encoder and decoder each:



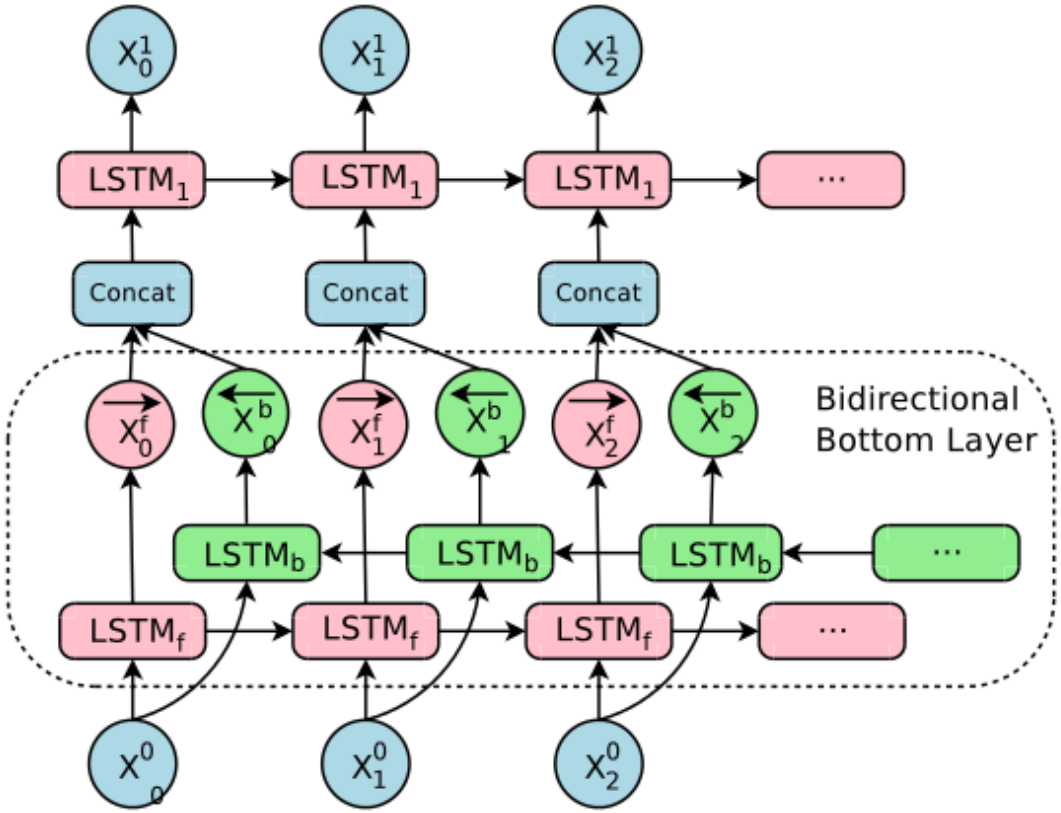
GOOGLE TRANSLATE

- September 2016: Wu et al., *Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation*:
  - but stacking LSTMs does not really work: 4-5 layers are OK, 8 layers don't work;
  - so they add residual connections between the layers, similar to (He, 2015):



GOOGLE TRANSLATE

- September 2016: Wu et al., *Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation*:
  - and it makes sense to make the bottom layer bidirectional in order to capture as much context as possible:



GOOGLE TRANSLATE

- September 2016: Wu et al., *Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation*:
- GNMT also uses two ideas for word segmentation:
  - *wordpiece model*: break words into wordpieces (with a separate model); example from the paper:

    `Jet makers feud over seat width with big orders at stake`

    becomes

    `_J et _makers _fe ud _over _seat _width _with _big _orders _at _stake`

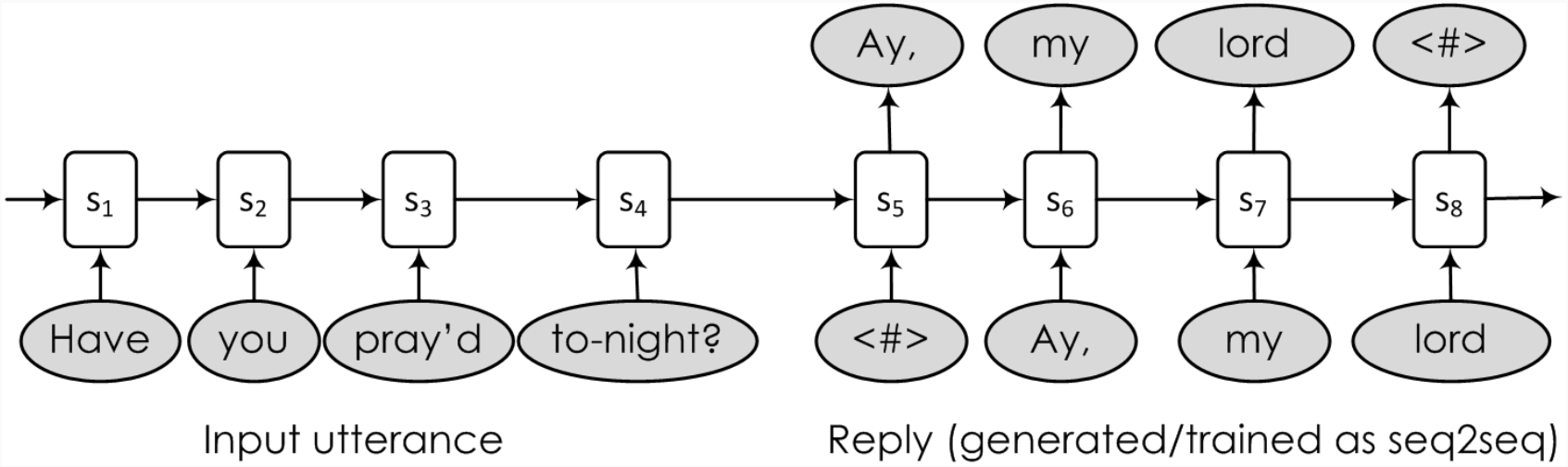
  - *mixed word/character model*: use word model but for out-of-vocabulary words convert them into characters (specifically marked so that they cannot be confused); example from the paper:

    `Miki` becomes `<B>M <M>i <M>k <E>i`

DIALOG AND CONVERSATION

DIALOG AND CONVERSATIONAL MODELS

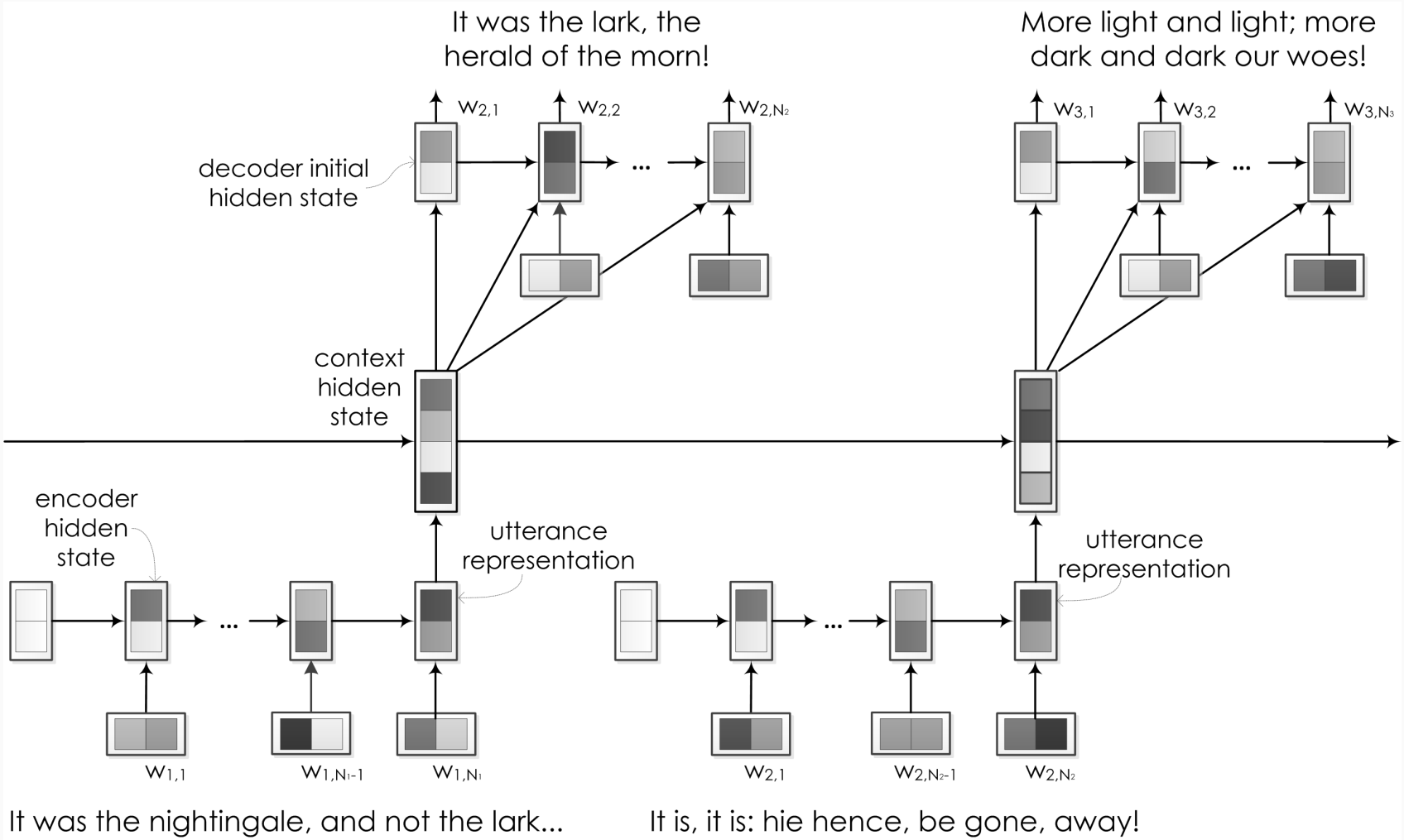
- Dialog models attempt to model and predict dialogue; conversational models actively talk to a human.
- Applications – automatic chat systems for business etc.
- Vinyals and Le (2015) use *seq2seq* (Sutskever et al. 2014):
  - feed previous sentences ABC as context to the RNN;
  - predict the next word of reply WXYZ based on the previous word and hidden state.



- Datasets: general (MovieSubtitles) or domain-specific (IT helpdesk).

- Hierarchical recurrent encoder decoder architecture (HRED); first proposed for query suggestion in IR (Sordoni et al. 2015), used for dialog systems in (Serban et al. 2015).
- The dialogue as a two-level system: a sequence of utterances, each of which is in turn a sequence of words. To model this two-level system, HRED trains:
  (1) *encoder* RNN that maps each utterance in a dialogue into a single utterance vector;
  (2) *context* RNN that processes all previous utterance vectors and combines them into the current context vector;
  (3) *decoder* RNN that predicts the tokens in the next utterance, one at a time, conditional on the context RNN.

- HRED architecture:



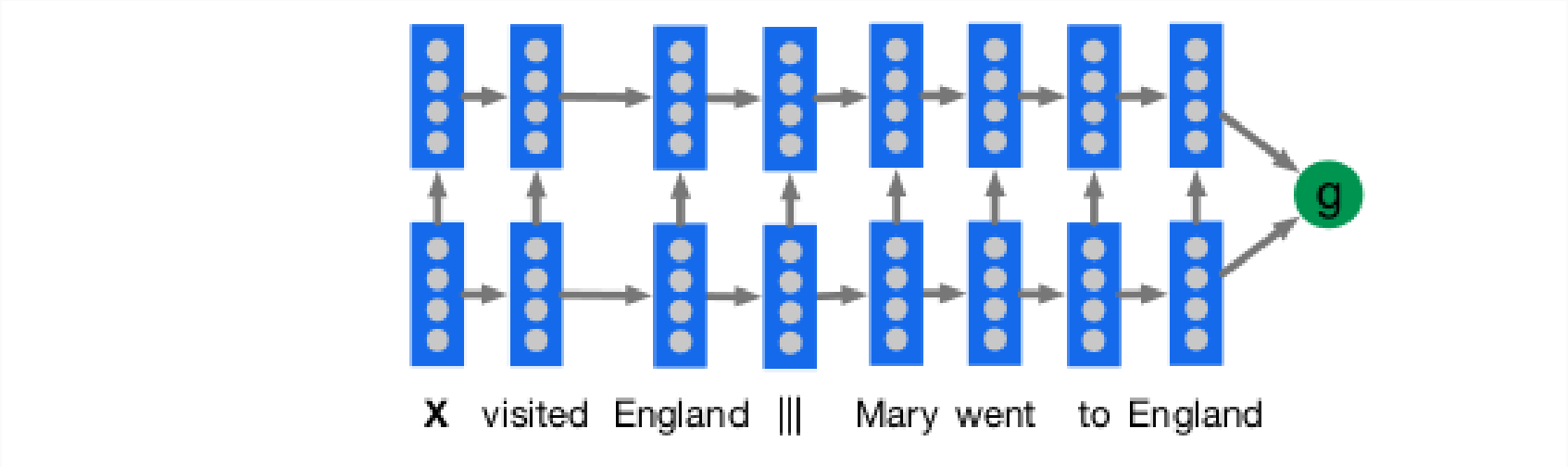
- Some recent developments:
  - (Li et al., 2016a) apply, again, reinforcement learning (DQN) to improve dialogue generation;
  - (Li et al., 2016b) add *personas* with latent variables, so dialogue can be more consistent (yes, it's the same Li);
  - (Wen et al., 2016) use *snapshot learning*, adding some weak supervision in the form of particular events occurring in the output sequence (whether we still want to say something or have already said it);
  - (Su et al., 2016) improve dialogue systems with online active reward learning, a tool from reinforcement learning.
- Generally, chatbots are becoming commonplace but it is still a long way to go before actual general-purpose dialogue.

TEACHING MACHINES TO READ

- (Hermann et al., 2015): «Teaching machines to read and comprehend» (Google DeepMind)
- A new way to construct a dataset for understanding by automated construction of (context, query, answer) triples from news items or similar texts.

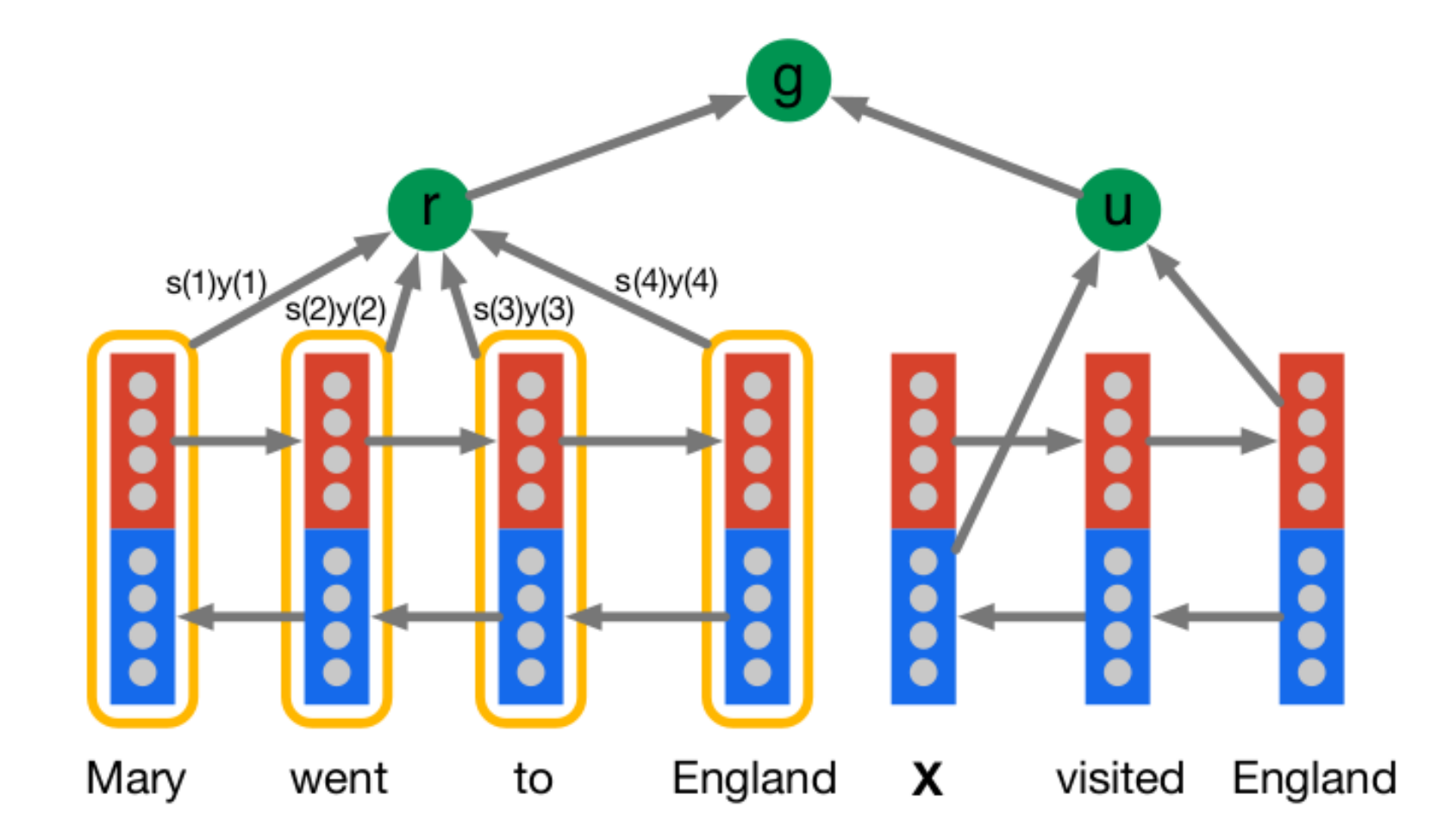
Original Version	Anonymised Version
Context	
The BBC producer allegedly struck by Jeremy Clarkson will not press charges against the "Top Gear" host, his lawyer said Friday. Clarkson, who hosted one of the most-watched television shows in the world, was dropped by the BBC Wednesday after an internal investigation by the British broadcaster found he had subjected producer Oisin Tymon "to an unprovoked physical and verbal attack." ...	the *ent381* producer allegedly struck by *ent212* will not press charges against the " *ent153* " host , his lawyer said friday . *ent212* , who hosted one of the most - watched television shows in the world , was dropped by the *ent381* wednesday after an internal investigation by the *ent180* broadcaster found he had subjected producer *ent193* " to an unprovoked physical and verbal attack . " ...
Query	
Producer **X** will not press charges against Jeremy Clarkson, his lawyer says.	producer **X** will not press charges against *ent212* , his lawyer says .
Answer	
Oisin Tymon	*ent193*

- The model is based on a deep network of LSTMs:

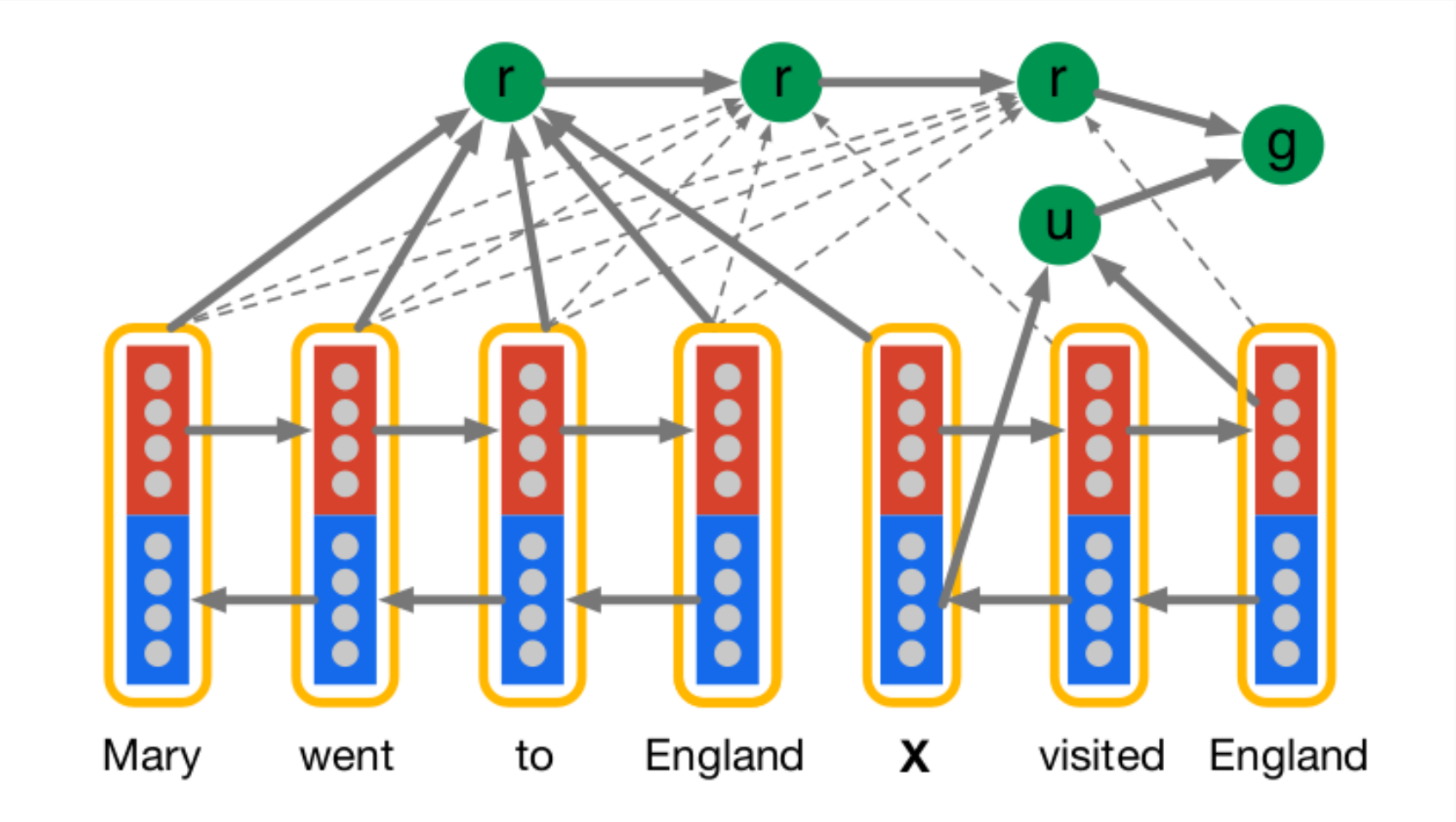


- But it does not work very well this way.

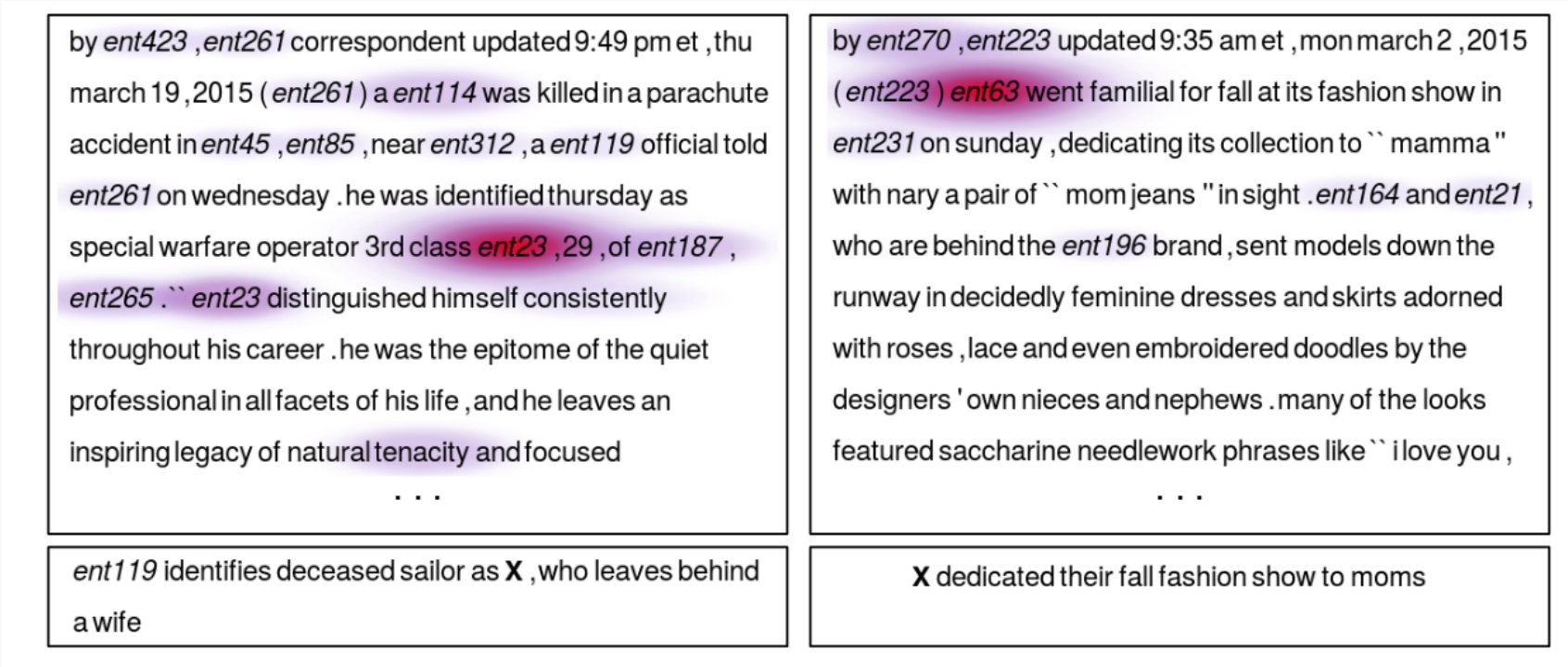
- Attentive Reader – train which part of the document to look at:



- Impatient Reader – re-read parts of the document as the query is processed:

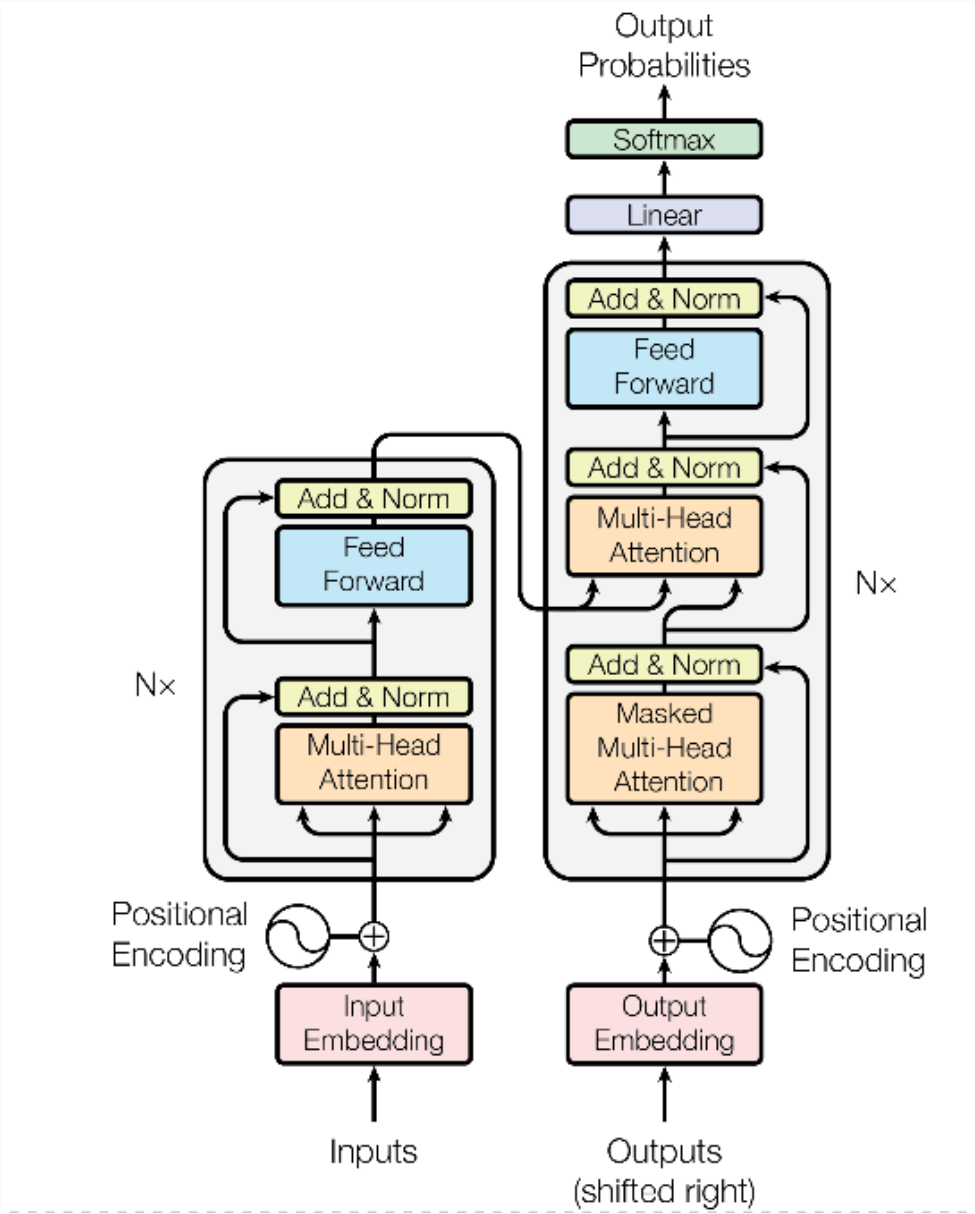


- Reasonable attention maps:



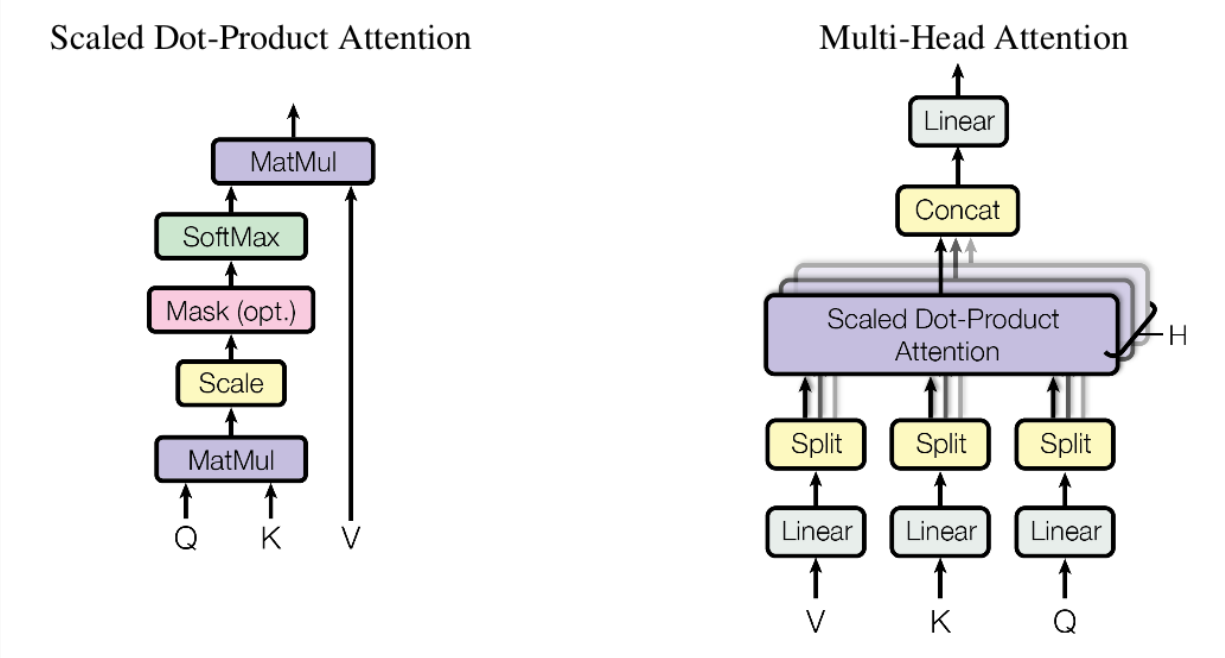
ATTENTION IS ALL YOU NEED

- June 2017: «Attention is all you need» (Vaswani et al., Google)



ATTENTION IS ALL YOU NEED

- Nothing but attention!
- Parallel attention maps are combined into matrices:



- Self-attention: each encoder position can “attend to” each position of the previous level.
- SMT results improve over state of the art, and training is 100x faster.

Thank you for your attention!

- It is hard to say what is “AI-complete” and what is not.
- The Turing test is perhaps not the best definition.
- But it would still be a huge step to teach computer models language.
- Unfortunately, we are still very far from decisive success here.
- But the problems are already in how to best encode common knowledge, common sense, things that make us human.
- What’s next? Let us see together.

DEEP LEARNING FOR RECOMMENDER SYSTEMS

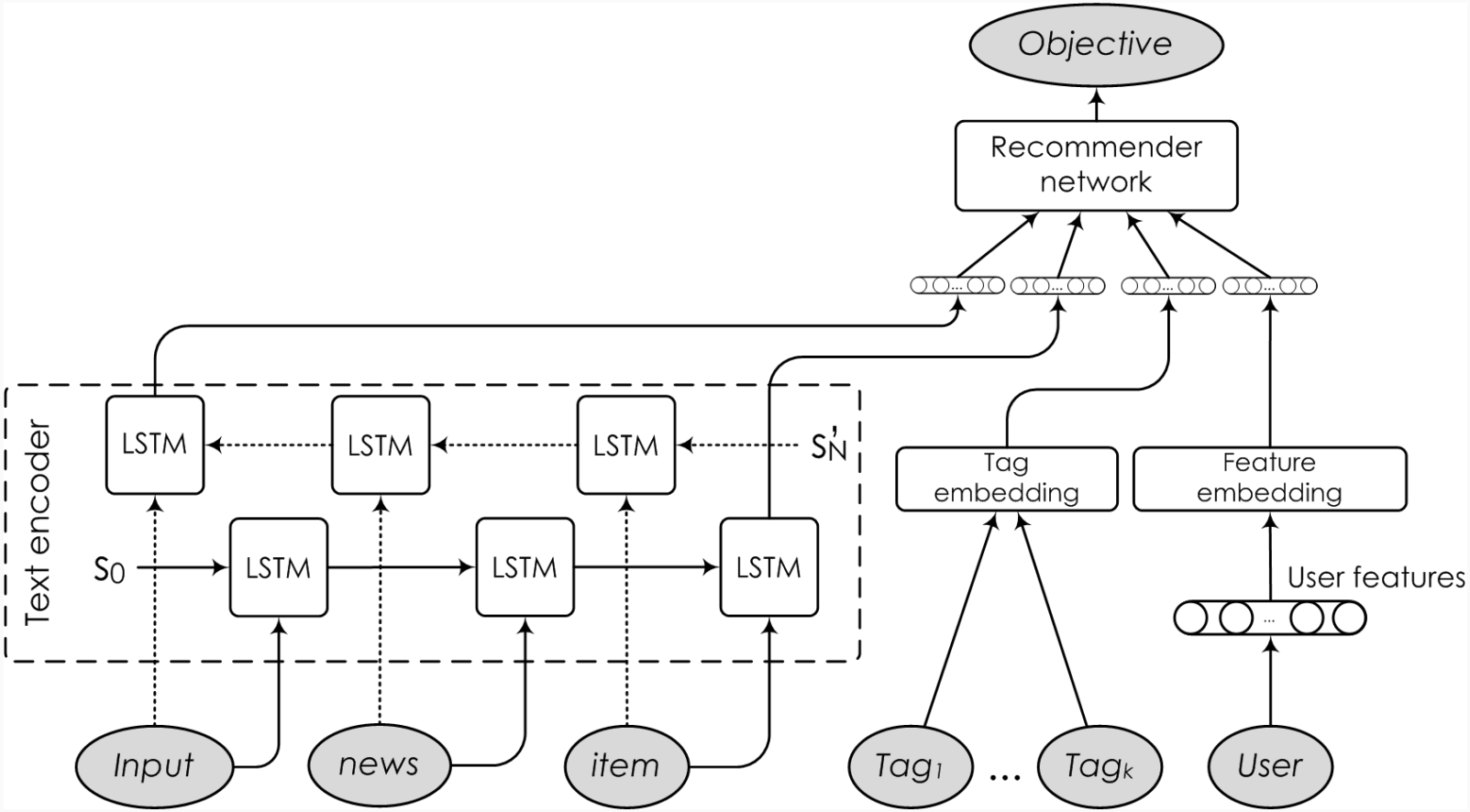
DL FOR RECOMMENDER SYSTEMS

- The deep learning revolution has led to important advances in nearly all fields of machine learning.
- But surprisingly few DL applications for recommender systems:
  - van den Oord et al. (2013) – content-based musical recommendations for Spotify;
  - Cheng et al. (2016) – Google Play recommender system based on "wide and deep" models;
  - Guo et al. (2017) – DeepFM, a neural network architecture that also uses the "wide and deep" idea;
  - Covington et al. (2016) – YouTube video recommendations that uses deep NNs first for candidate generation with coarse collaborative personalization and then for a more detailed re-ranking model.

DL FOR RECOMMENDER SYSTEMS

- We plan to develop new deep learning models that combine text understanding with recommender systems to be applied to items with text content (e.g., title, description, abstract, or the item itself).
- We plan to introduce novel neural architectures for:
  - extracting features from full-text items that can used for content-based recommendations;
  - actual recommendations that take into account textual features.

- Sample possible architecture:



- Based on these models, we plan to extract user profiles based on text content that will be readily interpretable and ready for use in other applications.
- We also plan to develop a software implementation for the models and its variations.
- We have access to a large dataset of user-generated general purpose texts published in a social network, which can be used for both training and testing (recommending texts to users).