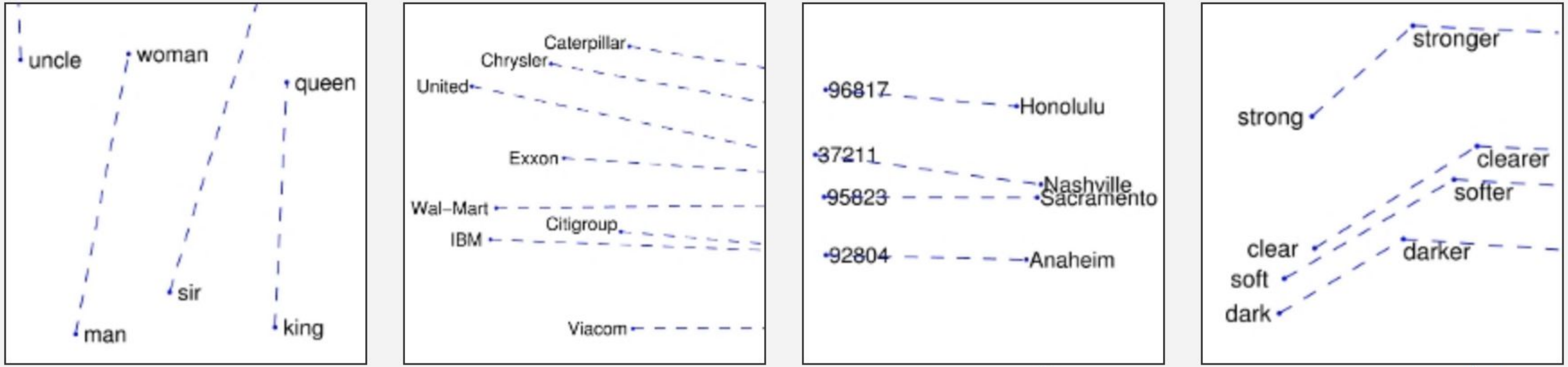
ELMo
Embeddings from Language Models

Mikhail Arkhipov - arkhipov@yahoo.com
Deep Learning Lab MIPT

Embeddings

- tl;dr - a vector representation of words
- Every word gets a trainable vector; surprisingly, embeddings create a "geometry" for words.
- First step of neural language modeling;
- Typically, 100-300 dimensional



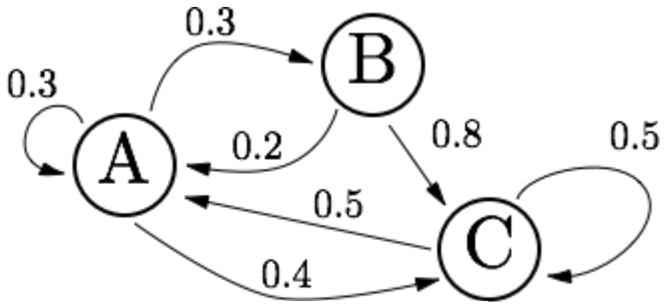
man - woman
company - ceo
city - zip code
comparative - superlative

Language modeling

By accurately assigning probability to a natural sequence (words or characters), you can improve:

- **Machine Translation:** p(strong tea) > p(powerful tea)
- **Speech Recognition:** p(speech recognition) > p(speech wreck ignition)
- **Question Answering / Summarization:**
  p(President X attended ...) is higher for X=Obama

Traditional approaches



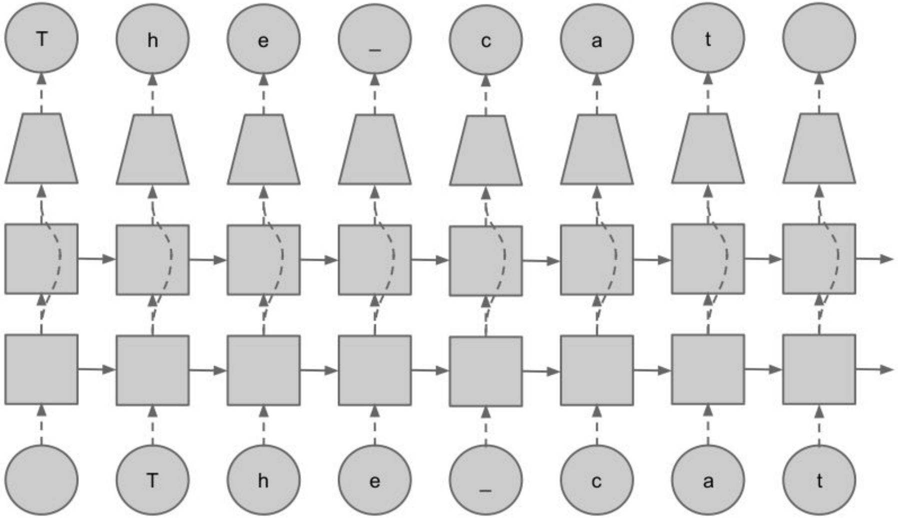
n-gram models and their adaptations:

$$\begin{aligned} &P(\mathrm{I}, \mathrm{saw}, \mathrm{the}, \mathrm{red}, \mathrm{house}) \\ \approx\; &P(\mathrm{I} \mid \langle s\rangle, \langle s\rangle) P(\mathrm{saw} \mid \langle s\rangle, I) P(\mathrm{the} \mid \mathrm{I}, \mathrm{saw}) P(\mathrm{red} \mid \mathrm{saw}, \mathrm{the}) P(\mathrm{house} \mid \mathrm{the}, \mathrm{red}) P(\langle /s\rangle \mid \mathrm{red}, \mathrm{house}) \end{aligned}$$

Issues:

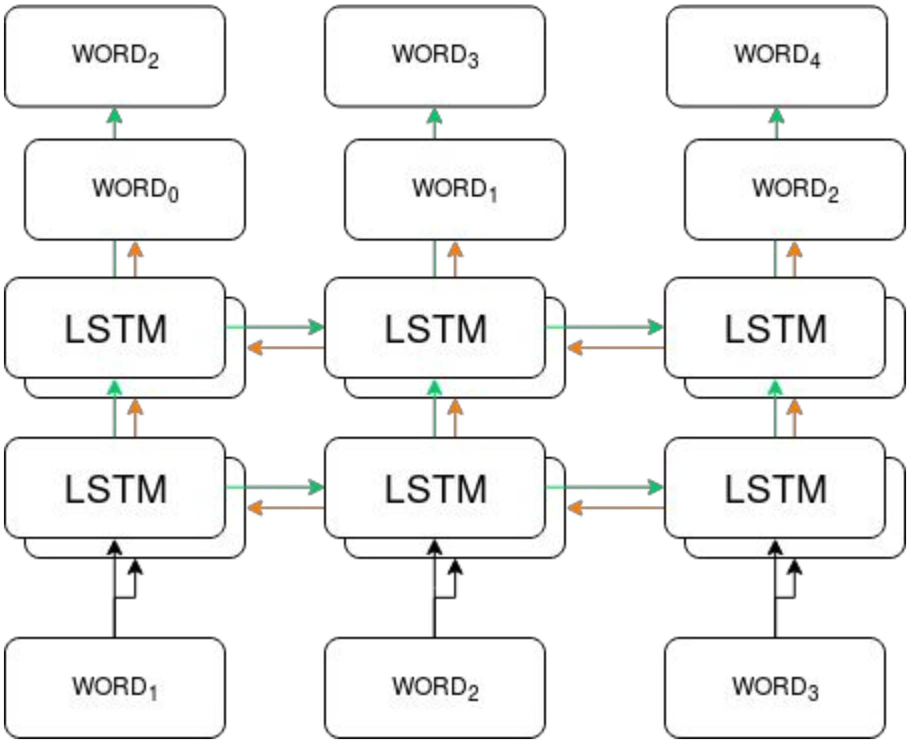
- What to do if n-gram has never been seen?
- How do you choose n for the n-grams?
- “Deep Learning is **amazing** because” v/s “Deep Learning is **awesome** because”

Language modeling with Recurrent Networks

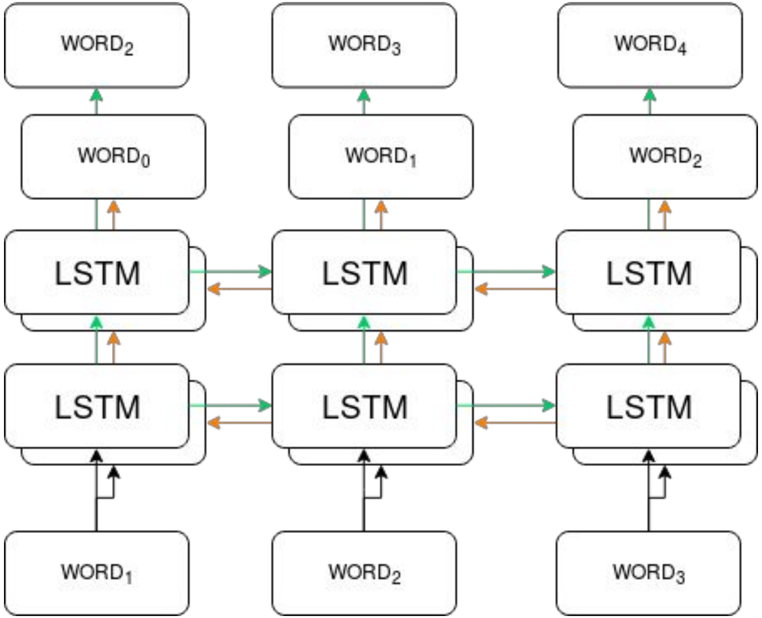


$$p(t_1, t_2, \ldots, t_N) = \prod_{k=1}^{N} p(t_k \mid t_1, t_2, \ldots, t_{k-1}).$$

Bi-directional Language Model

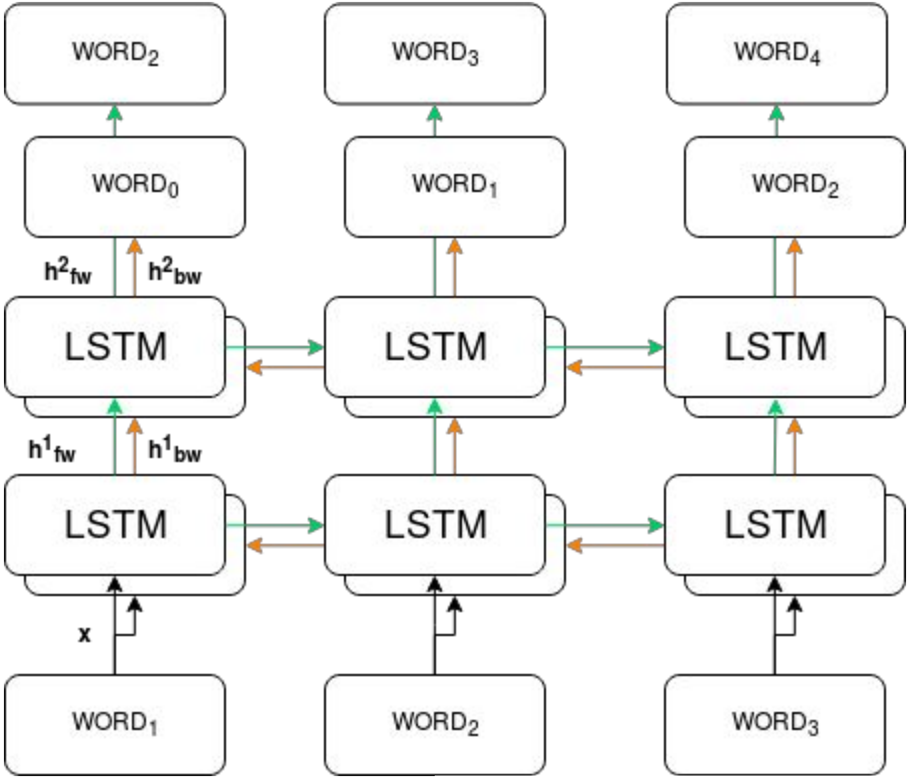


Bi-directional Language Model



$$\sum_{k=1}^{N} \left(\log p(t_k \mid t_1, \ldots, t_{k-1}; \Theta_x, \overrightarrow{\Theta}_{LSTM}, \Theta_s) \right.$$
$$\left. + \log p(t_k \mid t_{k+1}, \ldots, t_N; \Theta_x, \overleftarrow{\Theta}_{LSTM}, \Theta_s) \right) .$$

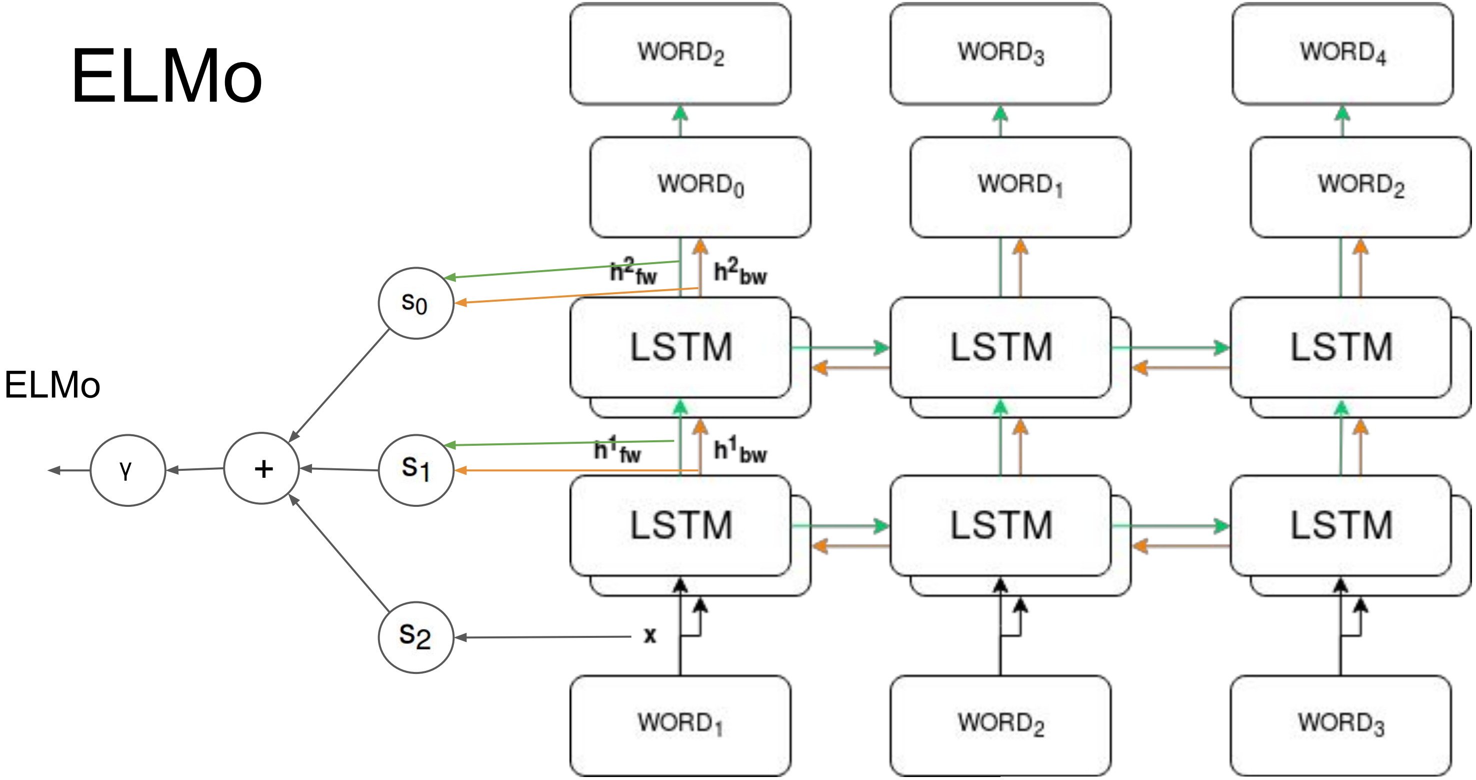
ELMo



$$\begin{aligned} R_k &= \{\mathbf{x}_k^{LM}, \overrightarrow{\mathbf{h}}_{k,j}^{LM}, \overleftarrow{\mathbf{h}}_{k,j}^{LM} \mid j = 1, \ldots, L\} \\ &= \{\mathbf{h}_{k,j}^{LM} \mid j = 0, \ldots, L\}, \end{aligned}$$

$$\mathbf{ELMo}_k^{task} = E(R_k; \Theta^{task}) = \gamma^{task} \sum_{j=0}^{L} s_j^{task} \mathbf{h}_{k,j}^{LM}.$$

ELMo



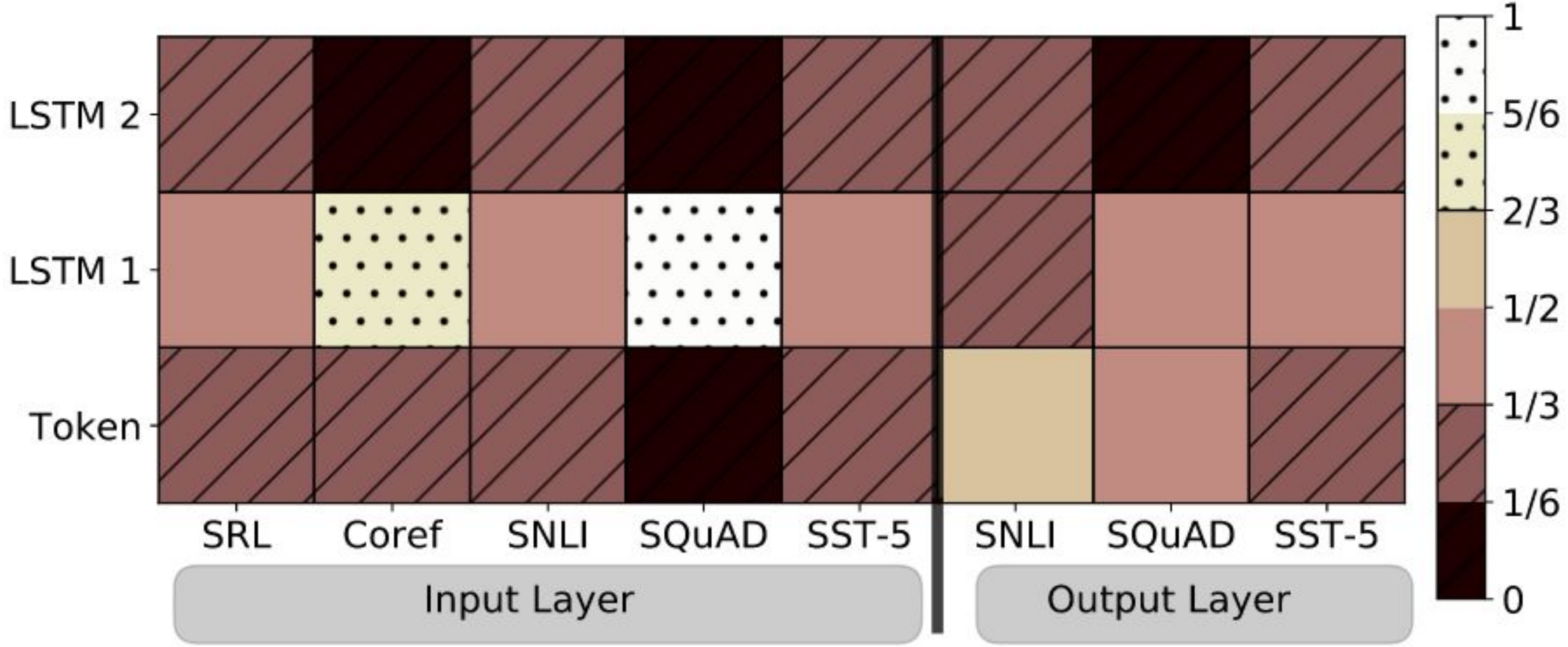
$$
\begin{aligned}
R_k &= \{\mathbf{x}_k^{LM}, \overrightarrow{\mathbf{h}}_{k,j}^{LM}, \overleftarrow{\mathbf{h}}_{k,j}^{LM} \mid j = 1, \ldots, L\} \\
&= \{\mathbf{h}_{k,j}^{LM} \mid j = 0, \ldots, L\},
\end{aligned}
$$

$$
\mathbf{ELMo}_k^{task} = E(R_k; \Theta^{task}) = \gamma^{task} \sum_{j=0}^{L} s_j^{task} \mathbf{h}_{k,j}^{LM}.
$$

ELMo results

TASK	PREVIOUS SOTA		OUR BASELINE	ELMO + BASELINE	INCREASE (ABSOLUTE/ RELATIVE)
SQuAD	Liu et al. (2017)	84.4	81.1	85.8	4.7 / 24.9%
SNLI	Chen et al. (2017)	88.6	88.0	88.7 ± 0.17	0.7 / 5.8%
SRL	He et al. (2017)	81.7	81.4	84.6	3.2 / 17.2%
Coref	Lee et al. (2017)	67.2	67.2	70.4	3.2 / 9.8%
NER	Peters et al. (2017)	91.93 ± 0.19	90.15	92.22 ± 0.10	2.06 / 21%
SST-5	McCann et al. (2017)	53.7	51.4	54.7 ± 0.5	3.3 / 6.8%

ELMo layer distribution



Spasibo