Deep Learning Basics

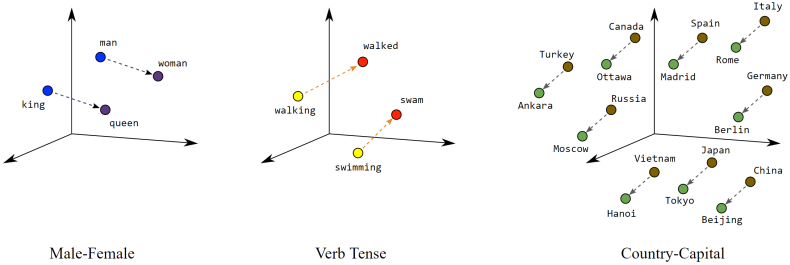
ELMo

Peters et al., 2018

Learning goals

- Understand the contextualization of word embeddings
- Get the intuition of feature-based Transfer Learning

RECAP: WORD VECTORS



Source: Google

- Information is encoded in (pre-)trained word embeddings
- Embeddings are used for tasks external to the training corpus

CONTEXTUALITY

1st Generation of neural embeddings are "context-free":

- Breakthrough paper (word2vec): ▸ Mikolov et al, 2013
- Followed by GloVe: ▸ Pennington et al, 2014
- FastText as extension of word2vec: ▸ Bojanowski et al, 2016

Why "Context-free"?

- Models learn *one single* embedding for each word
- Why could this possibly be problematic?
  - "The *default* setting of the function is xyz."
  - "The probability of *default* is rather high."
- Would be nice to have different embeddings for these two occurrences

CONTEXTUAL EMBEDDINGS

Source: Jay Alammar

EMBEDDINGS FROM LANGUAGE MODELS

- Bidirectional language model (LM)
- Combines a forward LM

$$p\left(w_1, w_2, \ldots, w_N\right) = \prod_{k=1}^{N} p\left(w_k | w_1, w_2, \ldots, w_{k-1}\right)$$

and a backward LM

$$p\left(w_1, w_2, \ldots, w_N\right) = \prod_{k=1}^{N} p\left(w_k | w_{k+1}, w_{k+2}, \ldots, w_N\right)$$

to arrive at the following loglikelihood:

$$\begin{gathered}\sum_{k=1}^{N} \left(\log p\left(w_k | w_1, \ldots, w_{k-1}; \Theta_x, \overrightarrow{\Theta}_{LSTM}, \Theta_s\right)\right. \\ \left. + \log p\left(w_k | w_{k+1}, \ldots, w_N; \Theta_x, \overleftarrow{\Theta}_{LSTM}, \Theta_s\right)\right)\end{gathered}$$

ELMO EMBEDDINGS

- Character-based (context-independent) token representations

$$\vec{x}_k^{LM}$$

- Two-layer biLSTM as main architecture:
  - Two context-dependent token representations *per layer*, i.e.

    $$\overrightarrow{\vec{h}}_{k,j}^{LM} \;\&\; \overleftarrow{\vec{h}}_{k,j}^{LM} \text{ for the } k\text{-th token in the } j\text{-th layer.}$$

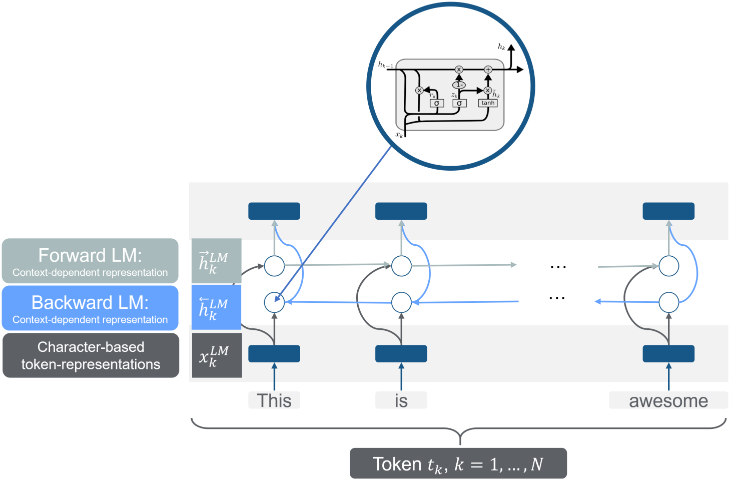
  - Four context-dependent token representations in total:

    $$\left\{ \overrightarrow{\vec{h}}_{k,j}^{LM}, \overleftarrow{\vec{h}}_{k,j}^{LM} | j = 1, 2 \right\}$$

- Five representations per token in total:

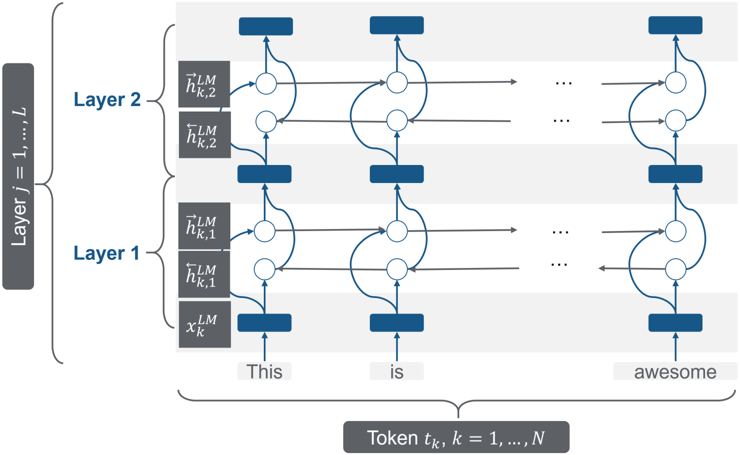
$$\begin{aligned} R_k &= \left\{ \vec{x}_k^{LM}, \overrightarrow{\vec{h}}_{k,j}^{LM}, \overleftarrow{\vec{h}}_{k,j}^{LM} | j = 1, \ldots, L \right\} \\ &= \left\{ \vec{h}_{k,j}^{LM} | j = 0, 1, 2 \right\} \end{aligned}$$

GRAPHICAL REPRESENTATION



▸ Source: Carolin Becker

GRAPHICAL REPRESENTATION



▸ Source: Carolin Becker

TASK ADAPTION

Including ELMo in downstream tasks:

- Calculate task-specific weights of all five representations:

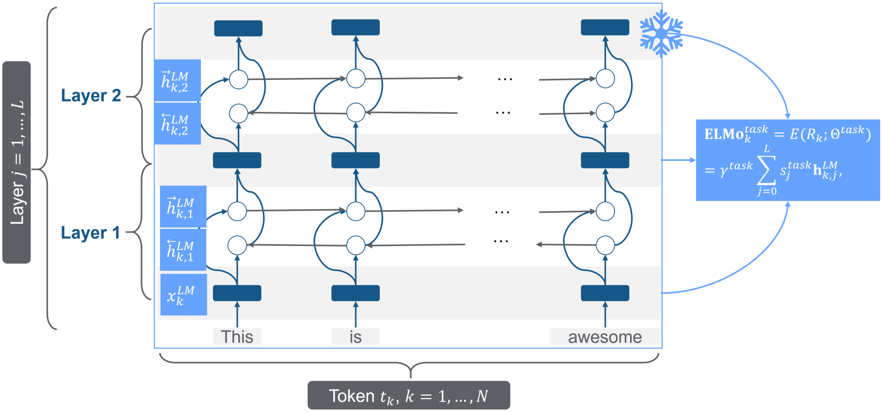
$$\mathbf{ELMo}_k^{task} = E\left(R_k; \Theta^{task}\right) = \gamma^{task} \sum_{j=0}^{L} s_j^{task} \vec{h}_{k,j}^{LM},$$

  where the $\vec{h}_{k,j}^{LM}$ are **not trainable** anymore.
- Trainable parameters during the adaption:
  - s_j^{task} are trainable (softmax-normalized) weights
  - γ^{task} is a trainable scaling parameter

Advantages over context free-embeddings:

- Task-specific model has access to *multiple* representations of each token
- Model learns to which degree to use the different representations depending on the task at hand

TASK ADAPTION



▸ Source: Carolin Becker

PERFORMANCE

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%

Table 1: Test set comparison of ELMo enhanced neural models with state-of-the-art single model baselines across six benchmark NLP tasks. The performance metric varies across tasks – accuracy for SNLI and SST-5; F_1 for SQuAD, SRL and NER; average F_1 for Coref. Due to the small test sizes for NER and SST-5, we report the mean and standard deviation across five runs with different random seeds. The "increase" column lists both the absolute and relative improvements over our baseline.

Source: Peters et al., 2018

SUMMARY

- Embeddings are (bidirectionally!) contextualized
  (as opposed to word2vec)
- Embeddings are *not* adapted to target domain/task
  (similar as for word2vec)
- Additional weights are learned for each downstream task
  (i.e. besides the embeddings, no shared knowledge across tasks)