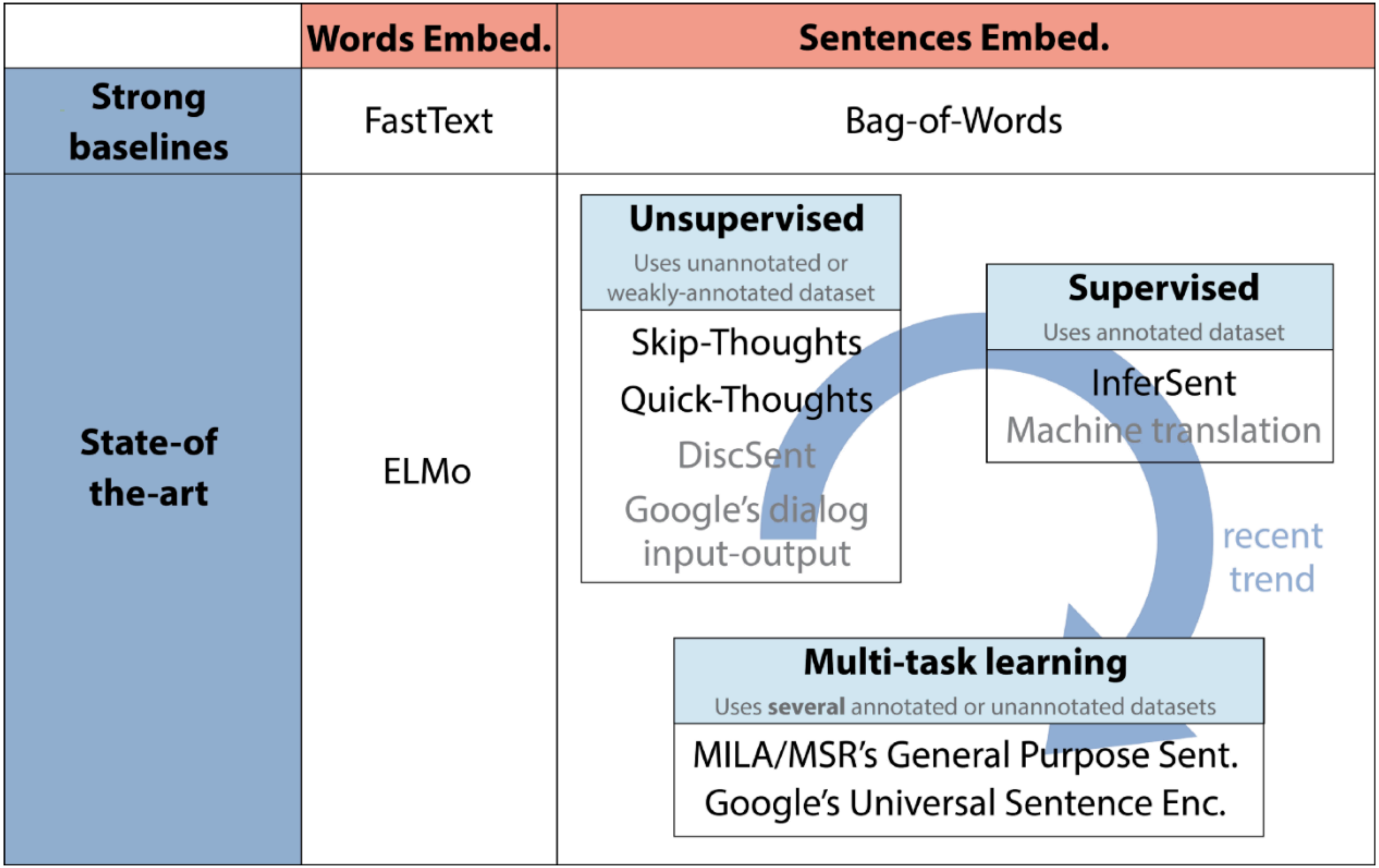
Математические методы анализа данных

Sentence encoders

Потапенко Анна Александровна

21 ноября 2018 г.

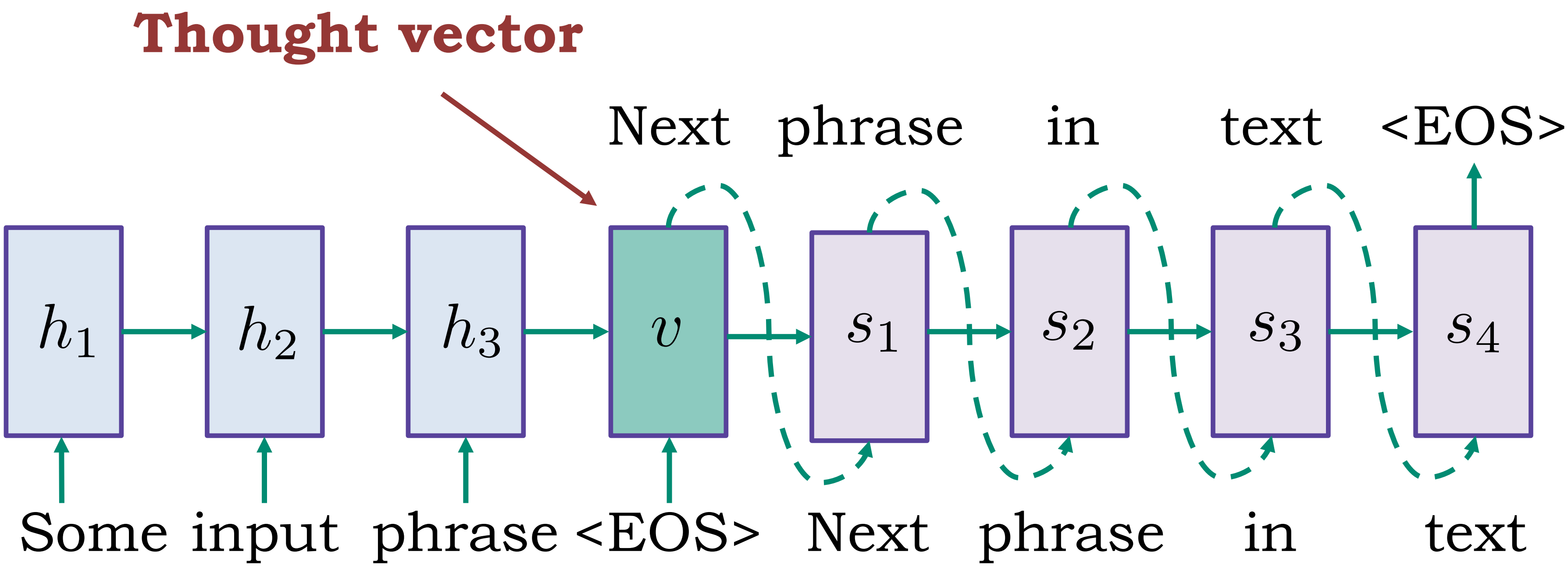
SOTA



https://medium.com/huggingface/universal-word-sentence-embeddings-ce48ddc8fc3a

Skip-thoughts (2015)

- Predicts next and previous sentences
- Encoder-decoder model (GRU or bi-GRU)



Kiros et. al. Skip-Thought Vectors, 2015, https://github.com/ryankiros/skip-thoughts.

SNLI dataset

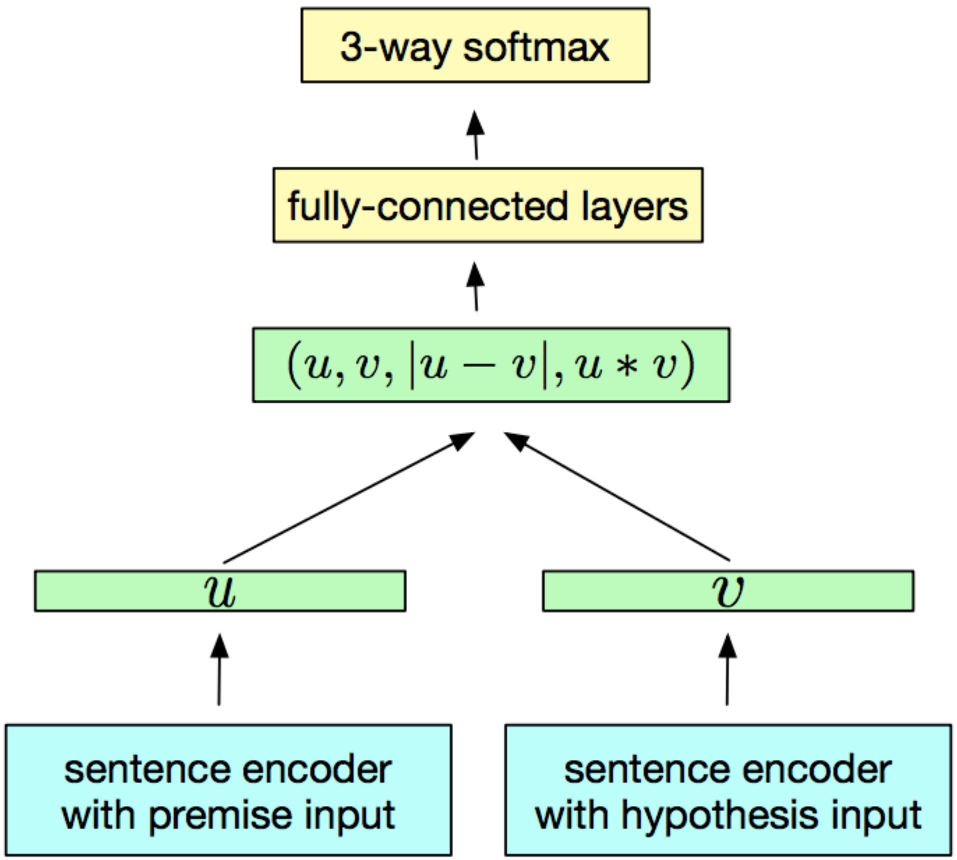
- Natural Language Inference task
- 570k human-written English sentence pairs
- Classification: *entailment, contradiction, neutral*
- Mechanical Turk judgements

Text	Judgments	Hypothesis
A man inspects the uniform of a figure in some East Asian country.	contradiction C C C C C	The man is sleeping
An older and younger man smiling.	neutral N N E N N	Two men are smiling and laughing at the cats playing on the floor.
A black race car starts up in front of a crowd of people.	contradiction C C C C C	A man is driving down a lonely road.
A soccer game with multiple males playing.	entailment E E E E E	Some men are playing a sport.
A smiling costumed woman is holding an umbrella.	neutral N N E C N	A happy woman in a fairy costume holds an umbrella.

Bowman et al. A large annotated corpus for learning natural language inference. https://nlp.stanford.edu/projects/snli/

InferSent (FAIR, 2017)

- Supervised training on SNLI corpus
- LSTM-encoder states aggregation methods:
  - First/last states
  - Pooling
  - Self-attention
  - Convolutions

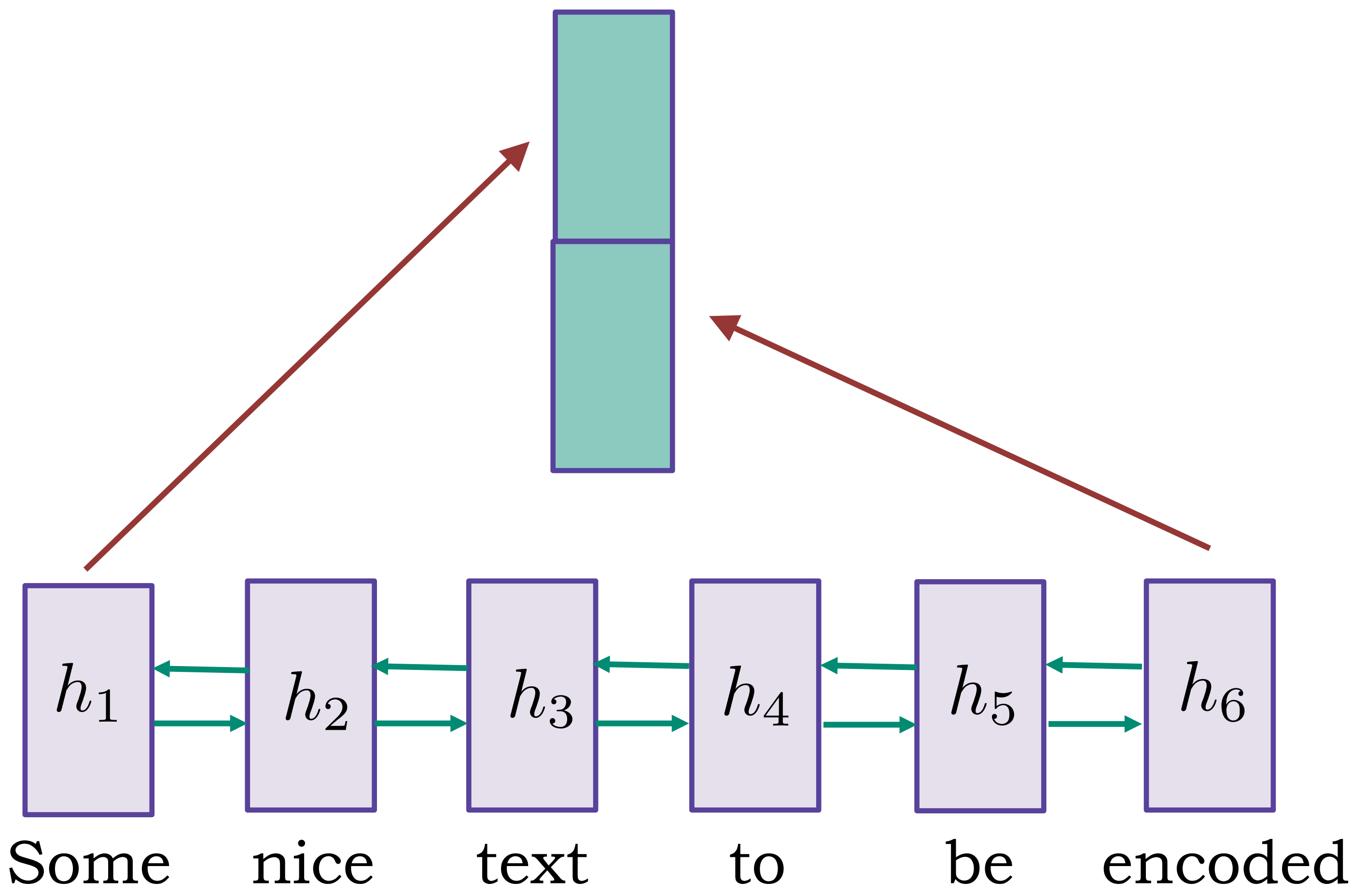


Paper: https://arxiv.org/pdf/1705.02364.pdf

Sentence encoders

- (bi-) LSTM / GRU

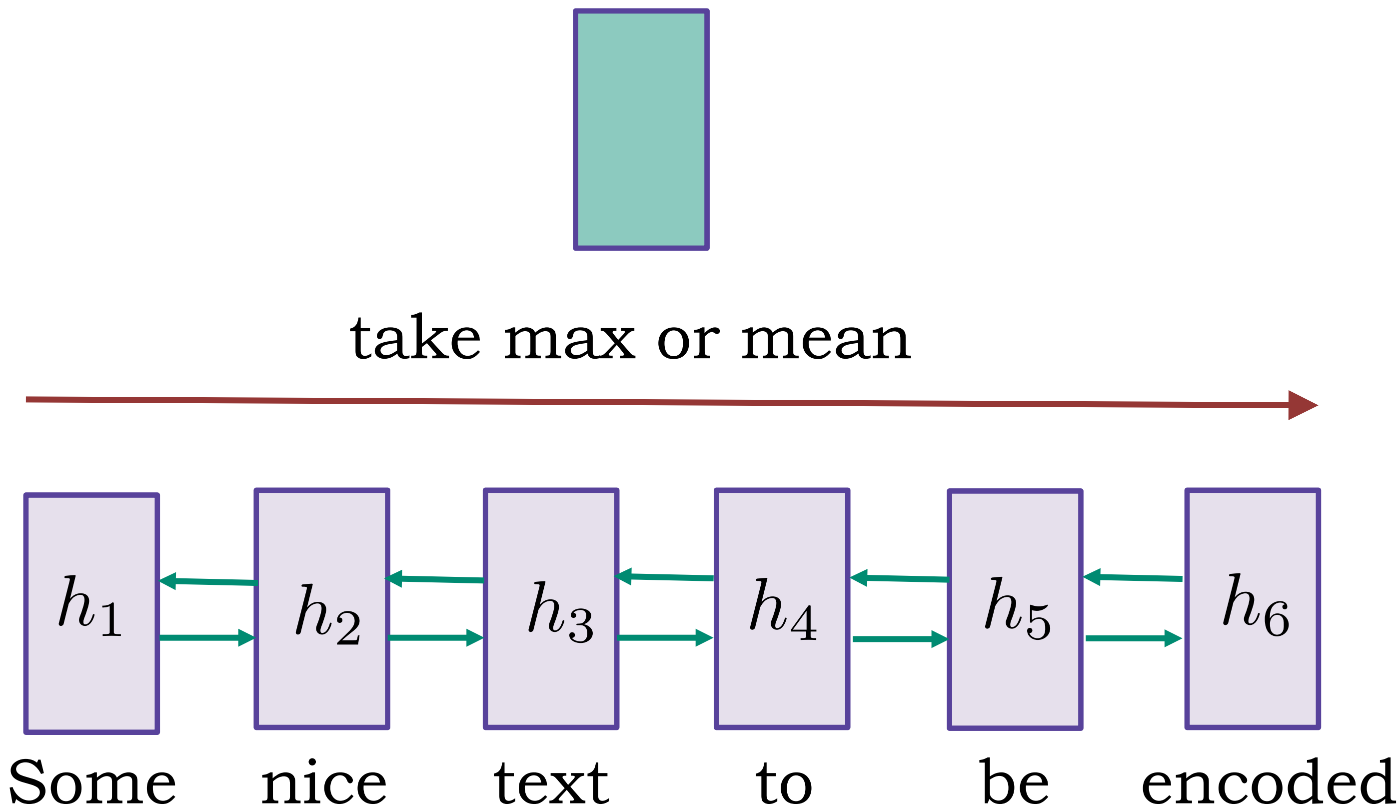
$$v = (h_1; h_n)$$



Pooling

- (bi-) LSTM / GRU
- Max/mean pooling

$$v = \frac{1}{n}\sum_{i=1}^{n} h_i$$

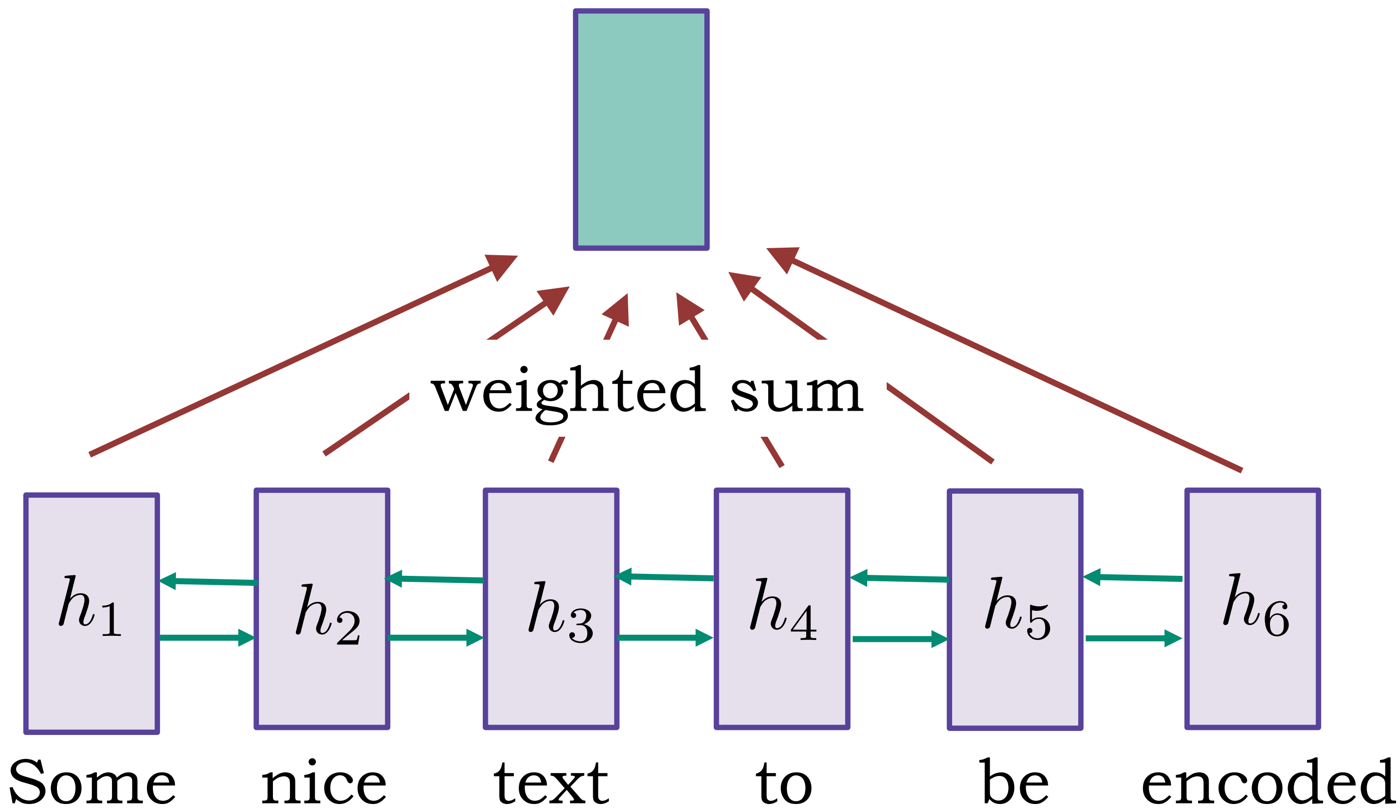


Inner-attention [Lin, 2017]

- (bi-) LSTM / GRU
- Self-attention

$$v = \sum_{i=1}^{n} a_i h_i$$

$$a = \mathrm{softmax}(w\,\mathrm{th}(W H^T))$$

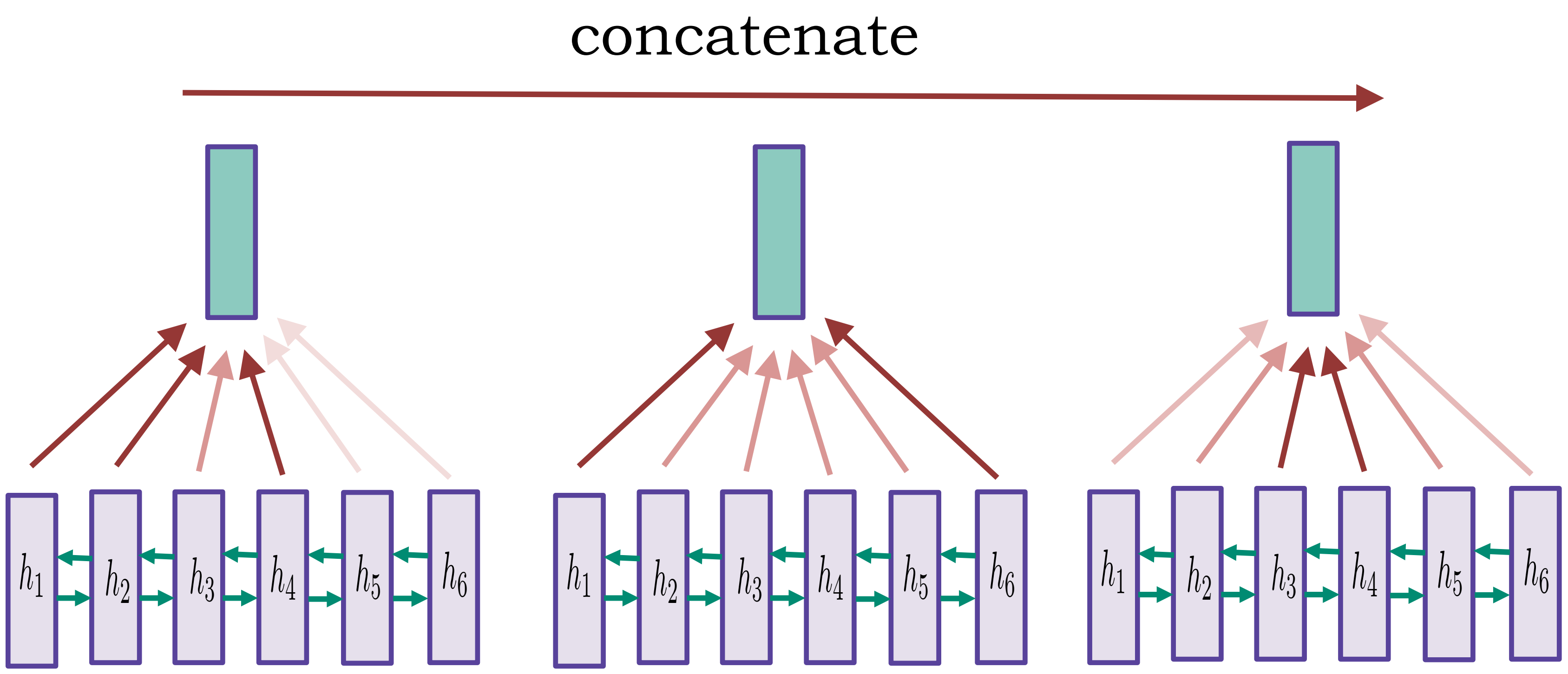


Inner-attention [Lin, 2017]

- (bi-) LSTM / GRU
- Self-attention
- Multiple heads!

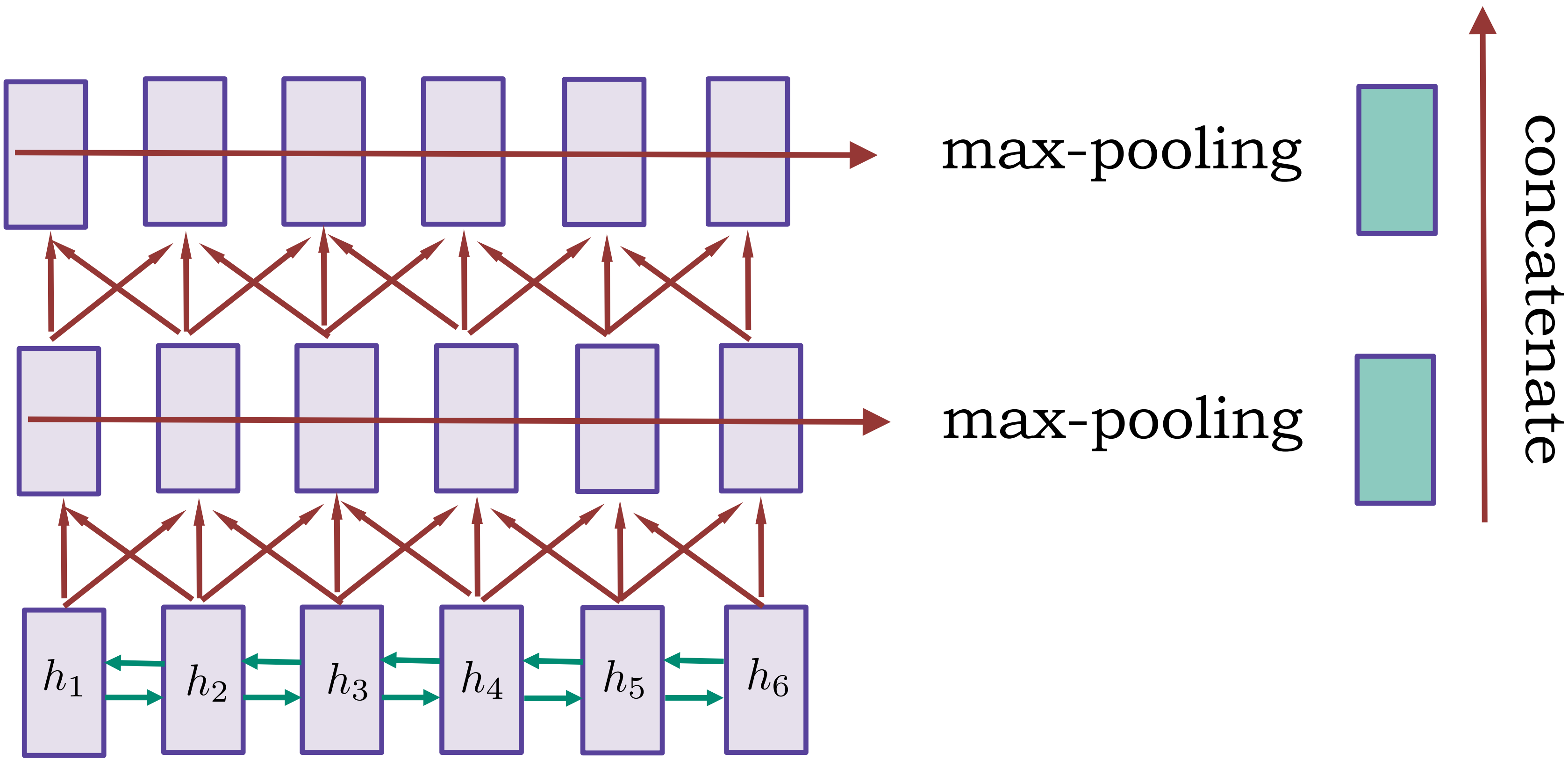
$$v = (v^1; v^2; \ldots; v^m)$$

$$v^k = \sum_{i=1}^{n} a_i^k h_i$$



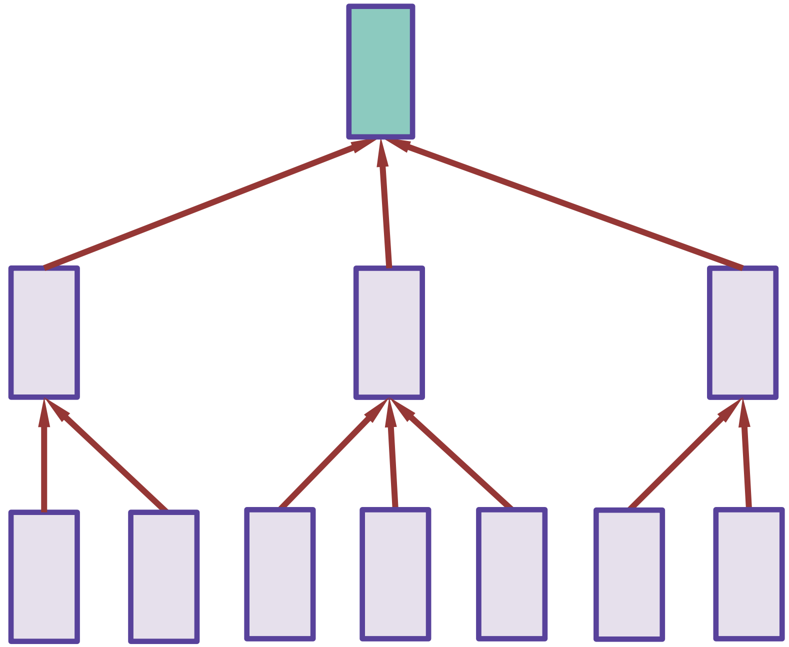
Convolutions

- (bi-) LSTM / GRU
- Hierarchical convolutions
- Multiple layers (4 in InferSent paper)



Dilated convolutions

- ❖ Here and before: LSTMs are not actually needed.
- ❖ Dilated convolutions: grow receptive field exponentially with linear increase in parameters.



ByteNet: Neural Machine Translation in Linear Time (2017)

WaveNet: A Generative Model for Raw Audio (2016)

Convolutions: parameters per layer

d – embeddings dimension;
n – sequence length;
k – filter size.

Parameters per layer:

- Usual convolution: O(k * d * d)

- Depth-wise convolution: O(k * d)

- Light-weight convolutions: O(k * d/b)

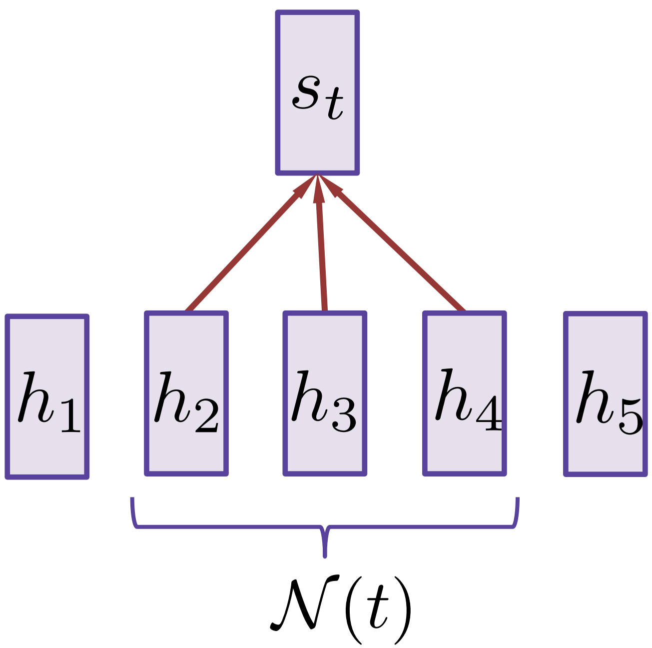
(To get time complexity, further multiply by n).

Usual convolutions

d – vector dimension; n – sequence length; k – filter size.

❖ Usual convolutions: O(k * d * d)

(k * d) weights for r^th element of t^th vector:



$$s_t^r = \sum_{i=1}^{k} \sum_{j=1}^{d} W_{ijr} h^j_{t+i-\lceil \frac{k}{2} \rceil}$$

Depth-wise convolutions

d – vector dimension; n – sequence length; k – filter size.

❖ Depth-wise convolutions: O(k * d)

j^{th} element of target depends on j^{th} element of source:

$$s_t^j = \sum_{i=1}^{k} W_{ij}\, h^j_{t+i-\lceil \frac{k}{2} \rceil}$$

❖ Light-weight convolutions: O(k * d/b)

convolution weights shared for blocks of size b:

$$s_t^j = \sum_{i=1}^{k} W_{i,j//b}\, h^j_{t+i-\lceil \frac{k}{2} \rceil}$$

Pay less attention [ICLR-2019]

d – vector dimension; n – sequence length; k – filter size.

❖ Dynamic convolutions: O(k * d/b * d)

Compute weights as a function of the current state h_t:

$$W_{il} = \mathrm{softmax}_i\left(\sum_{r=1}^{d} U_{i,l,r} h_t^r\right)$$

Use softmax to get weights normalized over positions.

- convolution weights depend on the current position t
- scales linearly in sequence length
- would not work for usual convolutions (k*d^3 params)

Paper: https://openreview.net/pdf?id=SkVhlh09tX

Attention is all you need [NIPS-2017]

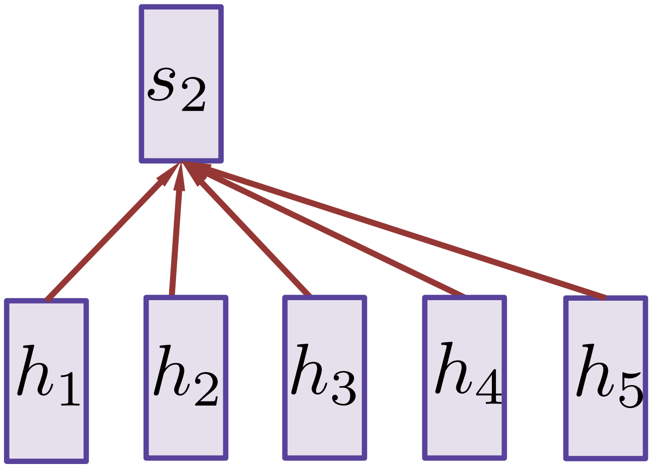
To find s_t given $[h_1, \ldots h_n]$:

- query: $q = W^q h_t$
- keys: $Y = W^k [h_1, \ldots h_n]$
- values: $V = W^v [h_1, \ldots h_n]$

$$s_t = \sum_{i=1}^{n} a_i \, v_i$$

$$a_i = \mathrm{softmax}\left(\frac{\langle q, y_i \rangle}{\sqrt{d}}\right)$$

- content-based
- quadratic in sequence length
- number of parameters?



Paper: https://arxiv.org/pdf/1706.03762.pdf

Multi-head: repeat and concatenate

To find s_t given $[h_1, \ldots h_n]$:

- query: $q = W^q h_t$
- keys: $Y = W^y [h_1, \ldots h_n]$
- values: $V = W^v [h_1, \ldots h_n]$

$$s_t = \sum_{i=1}^{n} a_i \, v_i$$

$$a_i = \operatorname{softmax}\left(\frac{\langle q, y_i \rangle}{\sqrt{d}}\right)$$

Number of parameters:

$m * d^y * d + m * d^y * d + m * d^v * d + m\, d^v * d = O(d^2)$

d – dimension of h and s (512)

m – number of heads (8)

d^y and d^v – dimensions of keys and values (64)

Convolutions vs self-attention

Layer Type	Complexity per Layer	Sequential Operations	Maximum Path Length
Self-Attention	$O(n^2 \cdot d)$	$O(1)$	$O(1)$
Recurrent	$O(n \cdot d^2)$	$O(n)$	$O(n)$
Convolutional	$O(k \cdot n \cdot d^2)$	$O(1)$	$O(log_k(n))$
Self-Attention (restricted)	$O(r \cdot n \cdot d)$	$O(1)$	$O(n/r)$

Depth-wise convolutions:

$$s_t = \sum_{\mathcal{N}(t)} W^T \odot H$$

Shared weights in blocks

Self-attention:

$$s_t = Ha$$

Multiple heads

Note: both need some operations along the depth (channels) dimension, e.g. linear or feed-forward:

$$f(h) = \mathrm{ReLu}(W_1 h + b_1) W_2 + b_2$$

Positional encoding

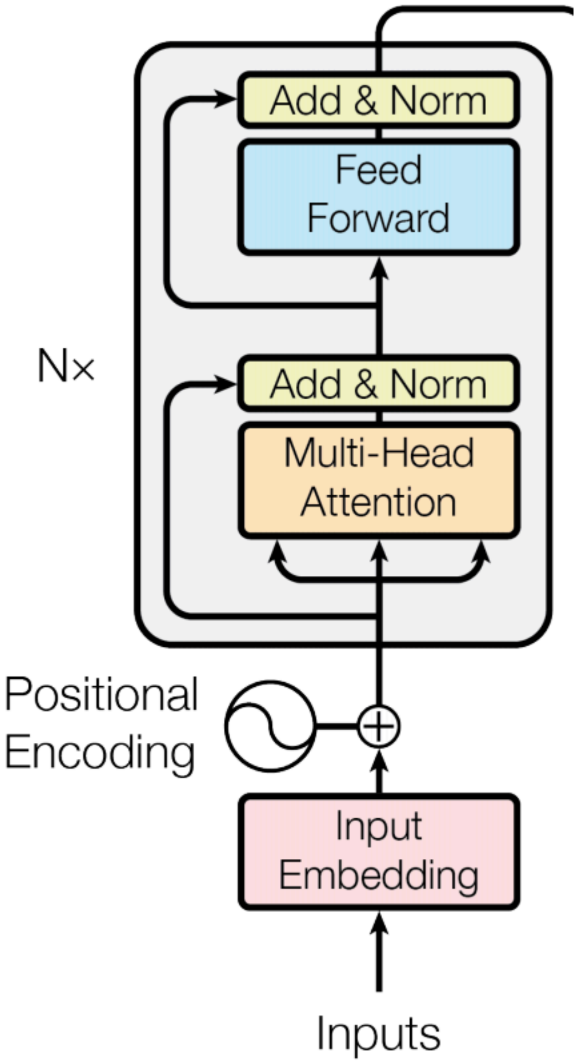
Четные компоненты вектора:

$$e_t^{(2j)} = \sin\left(\frac{t}{10000^{2j/d}}\right)$$

Нечетные компоненты вектора:

$$e_t^{(2j+1)} = \cos\left(\frac{t}{10000^{2j/d}}\right)$$

Для фиксированного сдвига k e_{t+k} выражается как линейная комбинация компонент e_t.



Annotated transformer: http://nlp.seas.harvard.edu/2018/04/03/attention.html

SentEval: 12 transfer tasks

- Binary and multi-class classification
  - sentiment analysis (MR, SST)
  - question-type (TREC)
  - product reviews (CR)
  - subjectivity/objectivity (SUBJ)
  - opinion polarity (MPQA)
- Entailment and semantic relatedness
  - SICK-E, SICK-R
- Paraphrase detection
  - Microsoft Research Paraphrase Corpus
- Caption-Image retrieval
  - COCO dataset

SentEval: 12 transfer tasks

name	N	task	C	examples
MR	11k	sentiment (movies)	2	”Too slow for a younger crowd , too shallow for an older one.” (neg)
CR	4k	product reviews	2	”We tried it out christmas night and it worked great .” (pos)
SUBJ	10k	subjectivity/objectivity	2	”A movie that doesn’t aim too high , but doesn’t need to.” (subj)
MPQA	11k	opinion polarity	2	”don’t want”; ”would like to tell”; (neg, pos)
TREC	6k	question-type	6	”What are the twin cities ?” (LOC:city)
SST	70k	sentiment (movies)	2	”Audrey Tautou has a knack for picking roles that magnify her [..]” (pos)

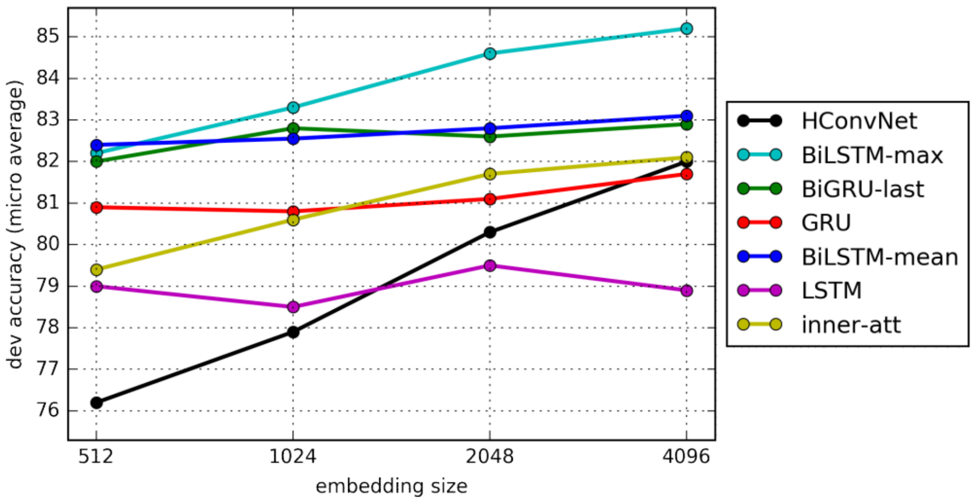
Table 1: **Classification tasks**. C is the number of class and N is the number of samples.

name	task	N	premise	hypothesis	label
SNLI	NLI	560k	”Two women are embracing while holding to go packages.”	”Two woman are holding packages.”	entailment
SICK-E	NLI	10k	A man is typing on a machine used for stenography	The man isn’t operating a stenograph	contradiction
SICK-R	STS	10k	”A man is singing a song and playing the guitar”	”A man is opening a package that contains headphones”	1.6
STS14	STS	4.5k	”Liquid ammonia leak kills 15 in Shanghai”	”Liquid ammonia leak kills at least 15 in Shanghai”	4.6

Table 2: **Natural Language Inference and Semantic Textual Similarity tasks**. NLI labels are contradiction, neutral and entailment. STS labels are scores between 0 and 5.

SentEval tool: https://github.com/facebookresearch/SentEval

Comparison of sentence embeddings (2017)



Model	dim	NLI dev	NLI test	Transfer micro	Transfer macro
LSTM	2048	81.9	80.7	79.5	78.6
GRU	4096	82.4	81.8	81.7	80.9
BiGRU-last	4096	81.3	80.9	82.9	81.7
BiLSTM-Mean	4096	79.0	78.2	83.1	81.7
Inner-attention	4096	82.3	82.5	82.1	81.0
HConvNet	4096	83.7	83.4	82.0	80.9
BiLSTM-Max	4096	**85.0**	**84.5**	**85.2**	**83.7**

InferSent paper: https://arxiv.org/pdf/1705.02364.pdf

Comparison of sentence embeddings (2017)

Model	MR	CR	SUBJ	MPQA	SST	TREC	MRPC	SICK-R	SICK-E	STS14
Unsupervised representation training (unordered sentences)										
Unigram-TFIDF	73.7	**79.2**	90.3	82.4	-	85.0	73.6/81.7	-	-	.58/.57
ParagraphVec (DBOW)	60.2	66.9	76.3	70.7	-	59.4	72.9/81.1	-	-	.42/.43
SDAE	74.6	78.0	90.8	86.9	-	78.4	**73.7**/80.7	-	-	.37/.38
SIF (GloVe + WR)	-	-	-	-	82.2	-	-	-	**84.6**	**.69**/ -
word2vec BOW†	77.7	79.8	90.9	88.3	79.7	83.6	72.5/81.4	0.803	78.7	.65/.64
fastText BOW†	78.3	81.0	**92.4**	87.8	**81.9**	84.8	**73.9/82.0**	0.815	78.3	.63/.62
GloVe BOW†	**78.7**	78.5	91.6	87.6	79.8	83.6	72.1/80.9	0.800	78.6	.54/.56
GloVe Positional Encoding†	78.3	77.4	91.1	87.1	80.6	83.3	72.5/81.2	0.799	77.9	.51/.54
BiLSTM-Max (untrained)†	77.5	**81.3**	89.6	**88.7**	80.7	**85.8**	73.2/81.6	**0.860**	83.4	.39/.48
Unsupervised representation training (ordered sentences)										
FastSent	70.8	78.4	88.7	80.6	-	76.8	72.2/80.3	-	-	**.63/.64**
FastSent+AE	71.8	76.7	88.8	81.5	-	80.4	71.2/79.1	-	-	.62/.62
SkipThought	76.5	80.1	93.6	87.1	82.0	<u>**92.2**</u>	**73.0/82.0**	**0.858**	82.3	.29/.35
SkipThought-LN	**79.4**	**83.1**	<u>**93.7**</u>	**89.3**	82.9	88.4	-	**0.858**	79.5	.44/.45
Supervised representation training										
CaptionRep (bow)	61.9	69.3	77.4	70.8	-	72.2	73.6/81.9	-	-	.46/.42
DictRep (bow)	76.7	78.7	90.7	87.2	-	81.0	68.4/76.8	-	-	**.67**/<u>**.70**</u>
NMT En-to-Fr	64.7	70.1	84.9	81.5	-	82.8	69.1/77.1	-		.43/.42
Paragram-phrase	-	-	-	-	79.7	-	-	0.849	83.1	<u>**.71**</u>/ -
BiLSTM-Max (on SST)†	(*)	83.7	90.2	89.5	(*)	86.0	72.7/80.9	0.863	83.1	.55/.54
BiLSTM-Max (on SNLI)†	79.9	84.6	92.1	**89.8**	83.3	**88.7**	75.1/82.3	<u>**0.885**</u>	<u>**86.3**</u>	.68/.65
BiLSTM-Max (on AllNLI)†	<u>**81.1**</u>	<u>**86.3**</u>	**92.4**	<u>**90.2**</u>	<u>**84.6**</u>	88.2	<u>**76.2/83.1**</u>	<u>**0.884**</u>	<u>**86.3**</u>	**.70/.67**

SOTA

	Words Embed.	Sentences Embed.
Strong baselines	FastText	Bag-of-Words
State-of the-art	ELMo	**Unsupervised** Uses unannotated or weakly-annotated dataset Skip-Thoughts Quick-Thoughts DiscSent Google's dialog input-output **Supervised** Uses annotated dataset InferSent Machine translation recent trend **Multi-task learning** Uses **several** annotated or unannotated datasets MILA/MSR's General Purpose Sent. Google's Universal Sentence Enc.

https://medium.com/huggingface/universal-word-sentence-embeddings-ce48ddc8fc3a

Universal Sentence Encoders (Google, 2018)

```
import tensorflow_hub as hub

embed = hub.Module("https://tfhub.dev/google/"
    "universal-sentence-encoder/1")

embedding = embed([
    "The quick brown fox jumps over the lazy dog."])
```

Two types of encoders:

- ❖ Transformer
- ❖ DAN (Deep Averaging Network)

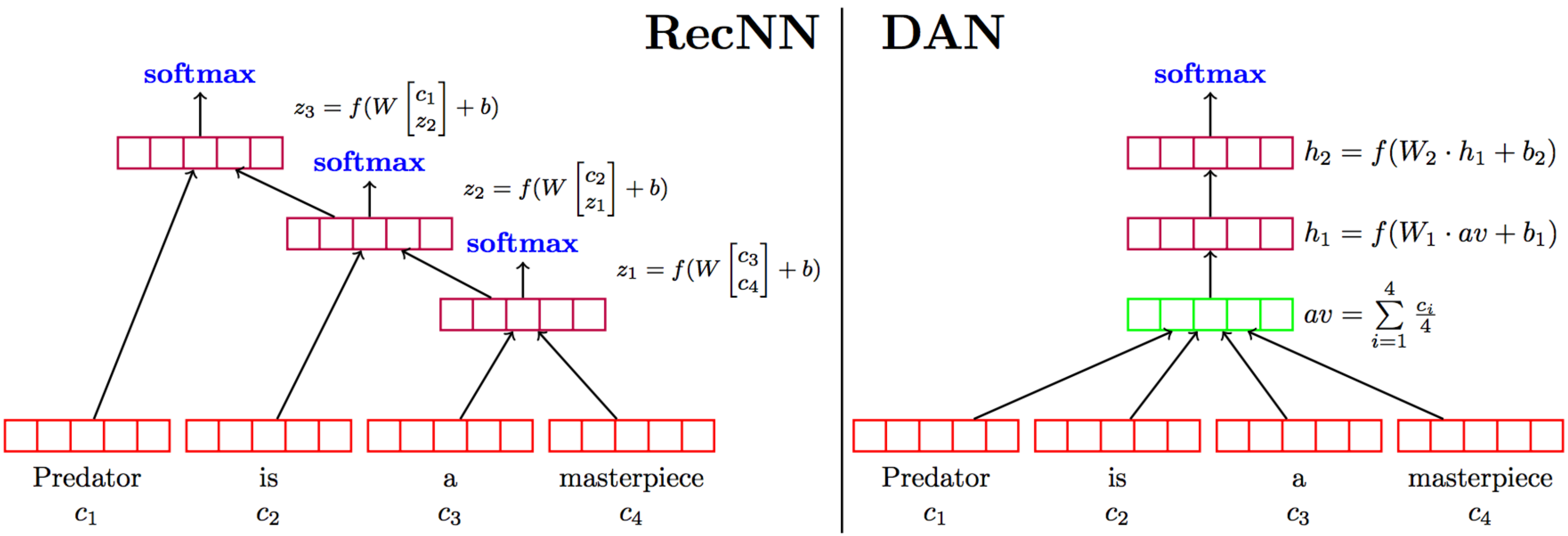
Lots of transfer tasks used for tuning the model.

USE paper: https://arxiv.org/abs/1803.11175

DAN: averaging + two layers

Syntax-aware models (out of scope of this lecture):

- Recursive NN: https://nlp.stanford.edu/~socherr/thesis.pdf
- TreeLSTM: https://www.aclweb.org/anthology/P15-1150
- DAG-LSTM: http://www.aclweb.org/anthology/N16-1106



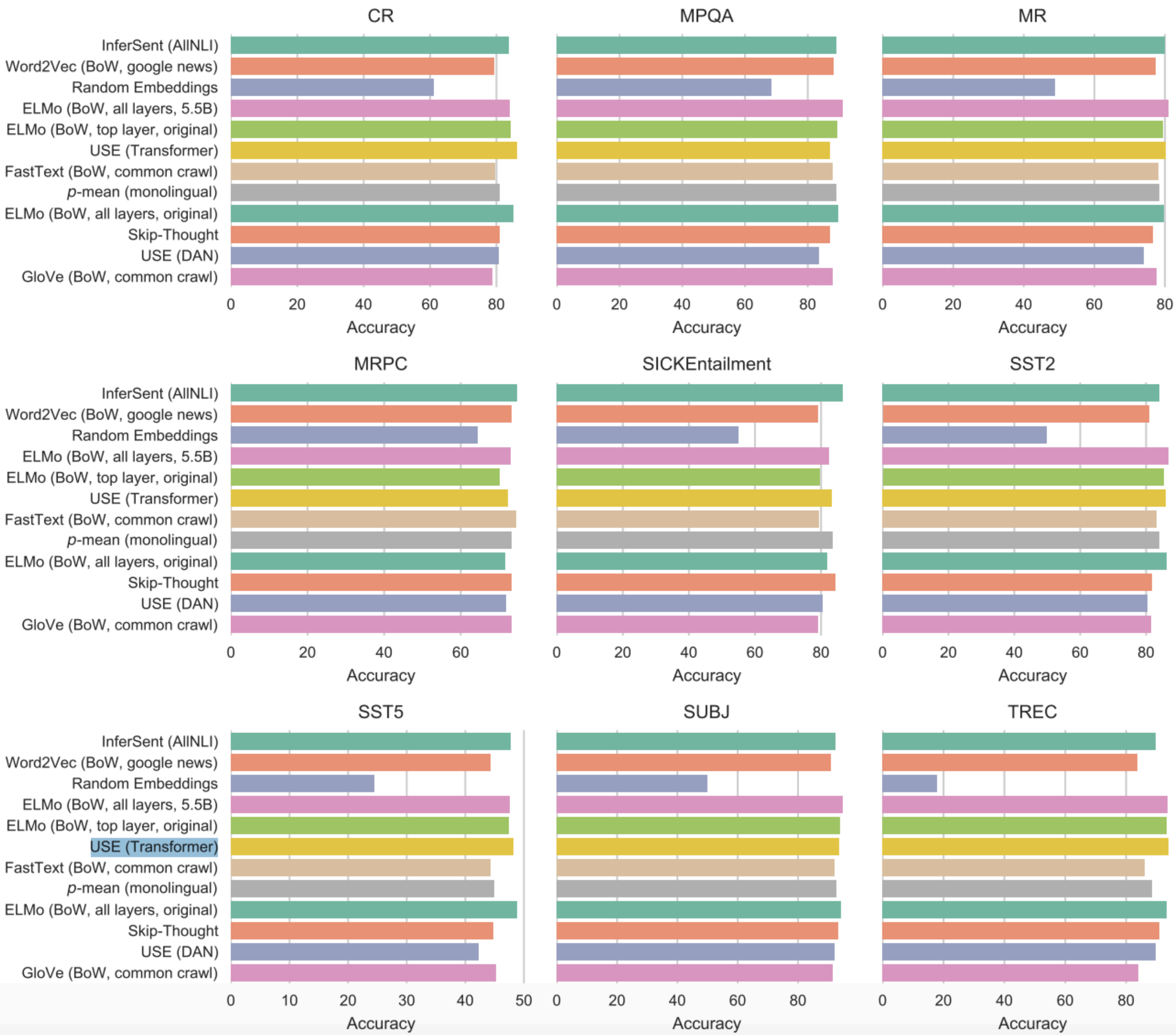
DAN paper: https://people.cs.umass.edu/~miyyer/pubs/2015_acl_dan.pdf

Comparison of sentence embeddings (2018)

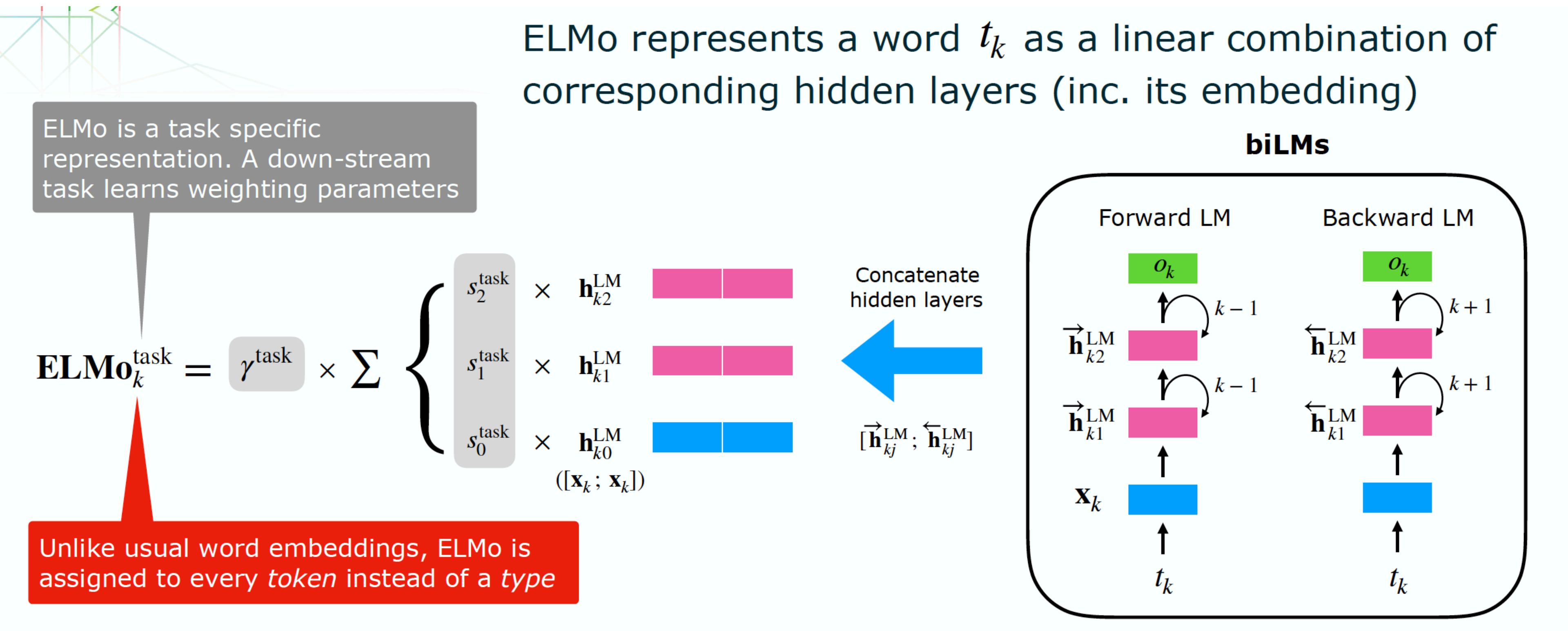
Approach	CR	MPQA	MR	MRPC	SICK-E	SST-2	SST-5	SUBJ	TREC
Baseline									
Random Embedding	61.16	68.41	48.75	64.35	54.94	49.92	24.48	49.83	18.00
Experiments									
ELMo (BoW, all layers, 5.5B)	83.95	**91.02**	**80.91**	72.93	82.36	**86.71**	47.60	**94.69**	93.60
ELMo (BoW, all layers, original)	85.11	89.55	79.72	71.65	81.86	86.33	**48.73**	94.32	93.40
ELMo (BoW, top layer, original)	84.13	89.30	79.36	70.20	79.64	85.28	47.33	94.06	93.40
Word2Vec (BoW, google news)	79.23	88.24	77.44	73.28	79.09	80.83	44.25	90.98	83.60
p-mean (monolingual)	80.82	89.09	78.34	73.22	83.52	84.07	44.89	92.63	88.40
FastText (BoW, common crawl)	79.63	87.99	78.03	74.49	79.28	83.31	44.34	92.19	86.20
GloVe (BoW, common crawl)	78.67	87.90	77.63	73.10	79.01	81.55	45.16	91.48	84.00
USE (DAN)	80.50	83.53	74.03	71.77	80.39	80.34	42.17	91.93	89.60
USE (Transformer)	**86.04**	86.99	80.20	72.29	83.32	86.05	48.10	93.74	**93.80**
InferSent (AllNLI)	83.58	89.02	80.02	**74.55**	**86.44**	83.91	47.74	92.41	89.80
SkipThought	81.03	87.06	76.60	73.22	84.33	81.77	44.80	93.33	91.00

Note: averaging ELMo (https://allennlp.org/elmo) context-aware word embeddings is really good!

Perone et al.: https://arxiv.org/pdf/1806.06259.pdf

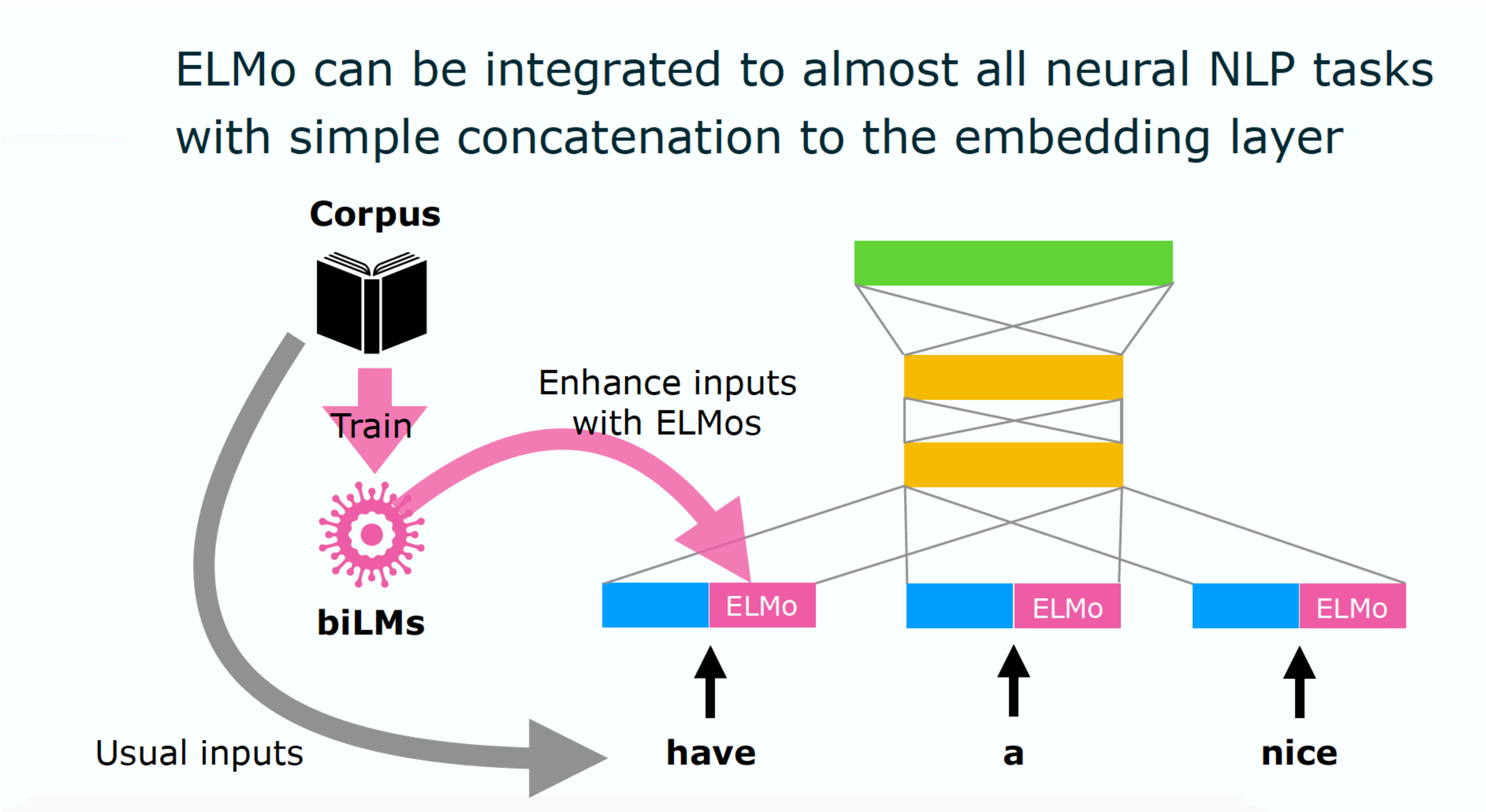

CR
MPQA
MR
InferSent (AllNLI)
Word2Vec (BoW, google news)
Random Embeddings
ELMo (BoW, all layers, 5.5B)
ELMo (BoW, top layer, original)
USE (Transformer)
FastText (BoW, common crawl)
p-mean (monolingual)
ELMo (BoW, all layers, original)
Skip-Thought
USE (DAN)
GloVe (BoW, common crawl)
0 20 40 60 80
Accuracy
MRPC
SICKEntailment
SST2
InferSent (AllNLI)
Word2Vec (BoW, google news)
Random Embeddings
ELMo (BoW, all layers, 5.5B)
ELMo (BoW, top layer, original)
USE (Transformer)
FastText (BoW, common crawl)
p-mean (monolingual)
ELMo (BoW, all layers, original)
Skip-Thought
USE (DAN)
GloVe (BoW, common crawl)
0 20 40 60
0 20 40 60 80
Accuracy
SST5
SUBJ
TREC
InferSent (AllNLI)
Word2Vec (BoW, google news)
Random Embeddings
ELMo (BoW, all layers, 5.5B)
ELMo (BoW, top layer, original)
USE (Transformer)
FastText (BoW, common crawl)
p-mean (monolingual)
ELMo (BoW, all layers, original)
Skip-Thought
USE (DAN)
GloVe (BoW, common crawl)
0 10 20 30 40 50
0 20 40 60 80
Accuracy

ELMo: model



[www.slideshare.nethttps://www.slideshare.net/mobile/shuntaroy/a-review-of-deep-contextualized-word-representations-peters-2018](https://www.slideshare.net/mobile/shuntaroy/a-review-of-deep-contextualized-word-representations-peters-2018)

ELMo: model



www.slideshare.nethttps://www.slideshare.net/mobile/shuntaroy/a-review-of-deep-contextualized-word-representations-peters-2018

ELMo: analysis

Many linguistic tasks are improved by using ELMo

	Task	Previous SOTA		Our baseline	ELMo + baseline	Increase (absolute/ relative)
Q&A	SQuAD	Liu et al. (2017)	84.4	81.1	85.8	4.7 / 24.9%
Textual entailment	SNLI	Chen et al. (2017)	88.6	88.0	88.7 ± 0.17	0.7 / 5.8%
Semantic role labelling	SRL	He et al. (2017)	81.7	81.4	84.6	3.2 / 17.2%
Coreference resolution	Coref	Lee et al. (2017)	67.2	67.2	70.4	3.2 / 9.8%
Named entity recognition	NER	Peters et al. (2017)	91.93 ± 0.19	90.15	92.22 ± 0.10	2.06 / 21%
Sentiment analysis	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.

[www.slideshare.nethttps://www.slideshare.net/mobile/shuntaroy/a-review-of-deep-contextualized-word-representations-peters-2018](https://www.slideshare.net/mobile/shuntaroy/a-review-of-deep-contextualized-word-representations-peters-2018)

ELMo: analysis

The higher layer seemed to learn semantics while the lower layer probably captured syntactic features

Word sense disambiguation

Model	F_1
WordNet 1st Sense Baseline	65.9
Raganato et al. (2017a)	69.9
Iacobacci et al. (2016)	**70.1**
CoVe, First Layer	59.4
CoVe, Second Layer	*64.7*
biLM, First layer	67.4
biLM, Second layer	*69.0*

Table 5: All-words fine grained WSD F_1. For CoVe and the biLM, we report scores for both the first and second layer biLSTMs.

PoS tagging

Model	Acc.
Collobert et al. (2011)	97.3
Ma and Hovy (2016)	97.6
Ling et al. (2015)	**97.8**
CoVe, First Layer	*93.3*
CoVe, Second Layer	92.8
biLM, First Layer	*97.3*
biLM, Second Layer	96.8

Table 6: Test set POS tagging accuracies for PTB. For CoVe and the biLM, we report scores for both the first and second layer biLSTMs.

www.slideshare.nethttps://www.slideshare.net/mobile/shuntaroy/a-review-of-deep-contextualized-word-representations-peters-2018

ELMo: analysis

The higher layer seemed to learn semantics while the lower layer probably captured syntactic features**???**

Most models preferred
“syntactic (probably)” features

Even in sentiment analysis

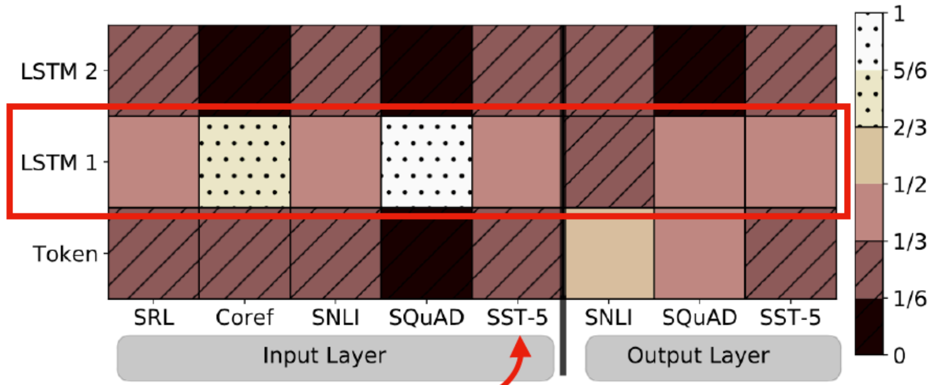


Figure 2: Visualization of softmax normalized biLM layer weights across tasks and ELMo locations. Normalized weights less then 1/3 are hatched with horizontal lines and those greater then 2/3 are speckled.

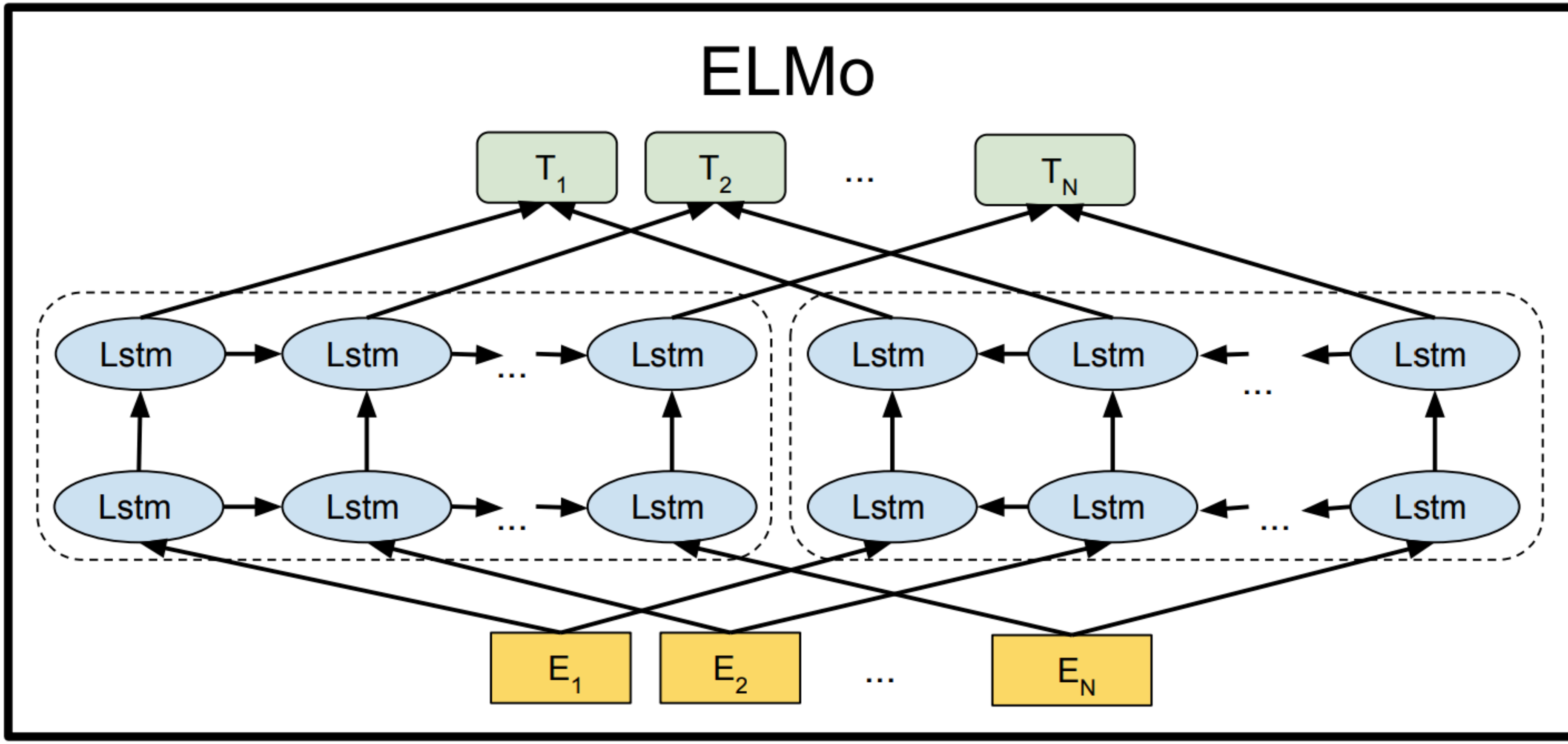
[www.slideshare.nethttps://www.slideshare.net/mobile/shuntaroy/a-review-of-deep-contextualized-word-representations-peters-2018](https://www.slideshare.net/mobile/shuntaroy/a-review-of-deep-contextualized-word-representations-peters-2018)

ELMo: conclusions

- Propose a new type of deep contextualised word representations (**ELMo**) that model:
  - Complex characteristics of word use (e.g., syntax and semantics)
  - How these uses vary across linguistic contexts (i.e., to model polysemy)
- Show that ELMo can improve existing neural models in various NLP tasks
- Argue that ELMo can capture more abstract linguistic characteristics in the higher level of layers

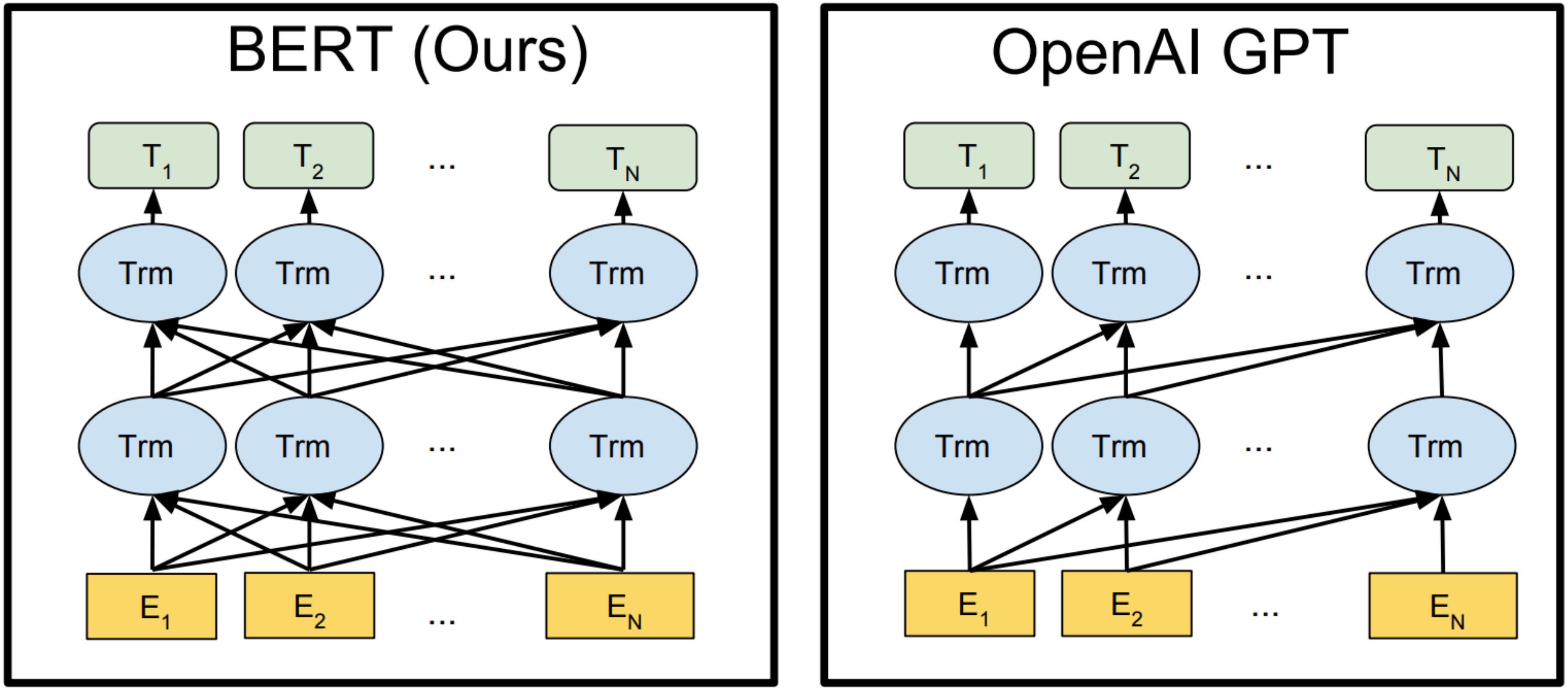
www.slideshare.nethttps://www.slideshare.net/mobile/shuntaroy/a-review-of-deep-contextualized-word-representations-peters-2018

BERT: bidirectional transformer [11 Oct 2018]



BERT paper: https://arxiv.org/pdf/1810.04805.pdf

BERT: bidirectional transformer [11 Oct 2018]



Trained in Masked Language Modeling setup.

BERT paper: https://arxiv.org/pdf/1810.04805.pdf

BERT: computation cost

"The cost of pre-training is actually somewhat more than moderate if you don't have access to a Cloud TPU pod :)

For example, OpenAI says that their 12 layer, 768-hidden Transformer took 1 month to train
on 8 P100s doing 40 epochs over an 800m word corpus.

BERT-Large is 24-layer, 1024-hidden and was trained for 40 epochs over a 3.3 billion word corpus. So maybe 1 year to train on 8 P100s?

16 Cloud TPUs is just a lot of computing power."

https://www.reddit.com/r/MachineLearning/comments/9nfqxz/r_%20bert_pretraining_of_deep_bidirectional/

Resume

Apart from LSTMs there are:

- convolutions (many different types)
- self-attention (transformers are hot now)
- recursive neural nets (syntax-aware)
- all types of (hierarchical) pooling techniques

Pre-trained word embeddings:

- ELMO, BERT (multi-lingual)

Pre-trained sentence embeddings:

- InferSent (and their SentEval tool)
- USE (via tf.hub + google.colab)