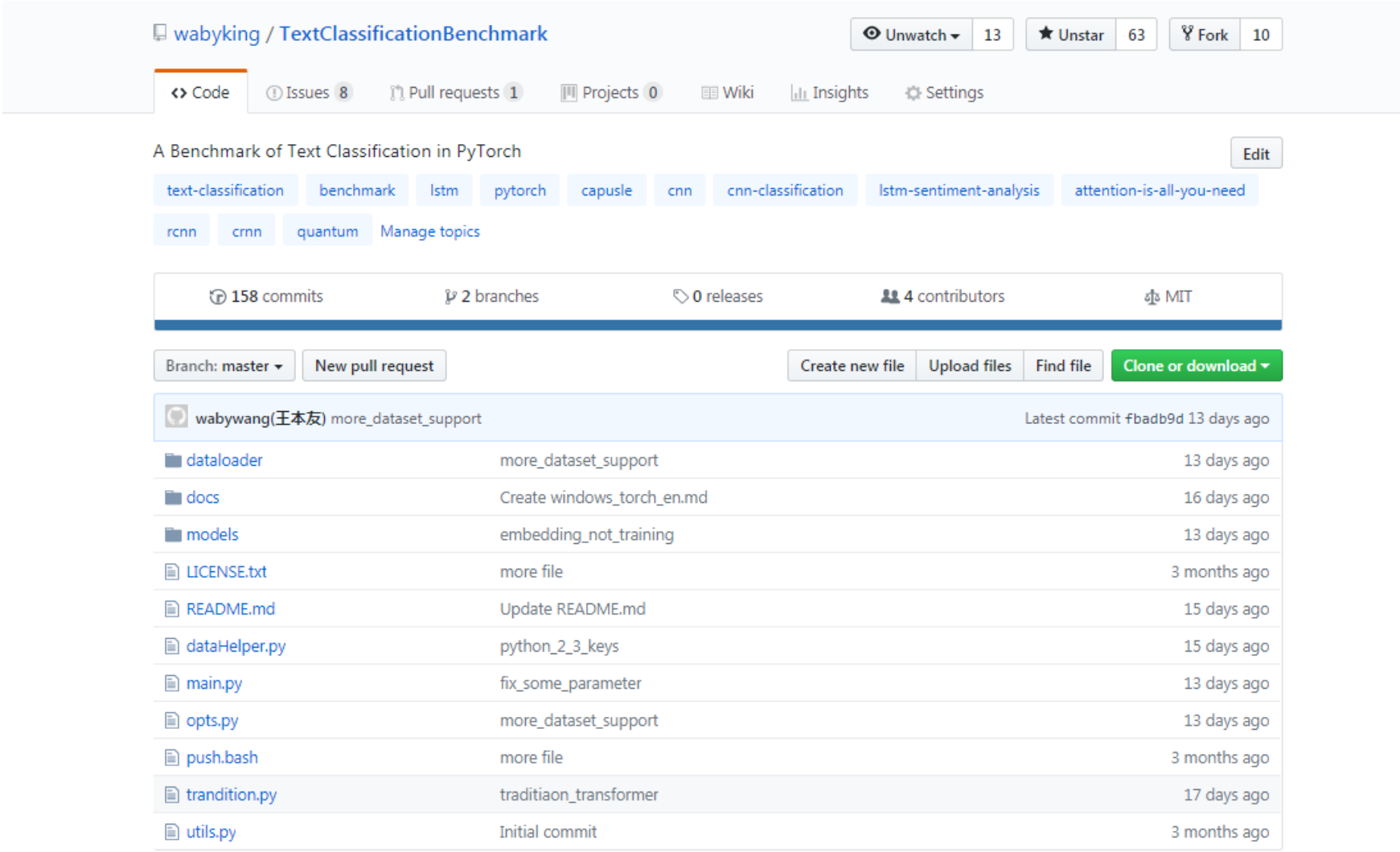
A survey and practice of Neural-network-based Textual representation

WabyWang,LilianWang,JaredWei,LoringLiu
Department of Social Network Operation,
Social Network Group,
Tencent

Wang B, Wang L, Wei Q, Wang Y, Liu L. TextZoo, a New Benchmark for Reconsidering Text Classification[J]. arXiv preprint arXiv:1802.03656, 2018.



welcome for any issues and contributions !!!

3256 lines

find . -name "*.py" -print | xargs wc -l

Cornell University Library

arXiv.org > cs > arXiv:1802.03656

Computer Science > Computation and Language

TextZoo, a New Benchmark for Reconsidering Text Classification

Benyou Wang, Li Wang, Qikang Wei, Lichun Liu

(Submitted on 10 Feb 2018 (v1), last revised 19 Mar 2018 (this version, v2))

TextZOO

A new Benchmark to Reconsidering Text Classification

Wang B, Wang L, Wei Q, Wang Y, Liu L. TextZoo, a New Benchmark for Reconsidering Text Classification[J]. arXiv preprint arXiv:1802.03656, 2018.

Can not do

- Can not directly deploy online
  - Implementing is easy, while design is what really challenging
- Can not tell you the precise hyper-parameter of your task
  - A fish or a fishing skill?
- Can not ensure to improve your performance
  - It depends on the scale of your supervised data

Highly depends on your data and task

- NLP features extraction Model
  - TFIDF is enough strong, e.s. long text
  - A Few pretrained Model
    - Glove/Word2vec only for initialization
    - No common-known CN embedding
    - No pretrained Model

- CV features extraction
  - SIFT or SIFT-like is not very strong.
  - pretrained ResNet from ImageNet

Zero-shot learning can hardly works in NLP, currently

Can do

- Easy to implement a model after talking
  - Talking is cheap, 10 lines a model.
- Directly support all the public dataset
  - Testing model
- Know how to design a DL model for NLP, not only text classification
  - A fishing skill

Contents

- Brief Introduction of TextZoo
- Why text classification？
- General Overview of Text Classification
- Overview of Text Classification in Neural Network approach.
- Architecture of TextZoo
- Conclusions

Contents

- Brief Introduction of TextZoo
- Why text classification？
- General Overview of Text Classification
- Overview of Text Classification in Neural Network approach.
- Architecture of TextZoo
- Conclusions

TextZoo

- Text Classification
  - Sentimental
  - Topic
  - Spam filter
  - …
- A benchmark
  - 20 Dataset
  - 20 Models
- PyTorch
  - Life is short, I use PyTorch(Python)

Models

- ✓FasText
- ✓CNN (Kim CNN, Multi-Layer CNN, Multi-perspective CNN, Inception CNN)
- ✓LSTM (BILSTM, StackLSTM, LSTM with Attention)
- ✓Hybrids between CNN and RNN (RCNN, C-LSTM)
- ✓Attention (Self Attention / Quantum Attention)
- ✓Transformer - Attention is all you need
- ✓Capsule
- ✓Quantum-inspired NN
- ➢ConS2S
- ➢Memory Network

Datasets

- **IMDB**
- **MR**
- CR
- MPQA
- SST1
- SST2
- Subj
- TREC

Contents

Supervised tasks in NLP

- Classification: assigning a label to a string

$$s \rightarrow c$$

- Matching: matching two strings

$$s, t \rightarrow \mathbf{R}^{+}$$

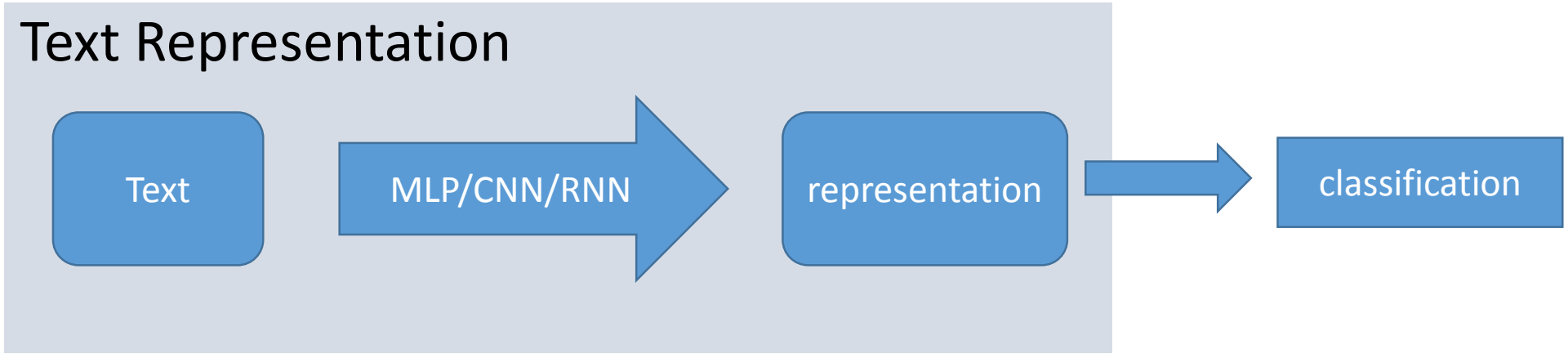
- Translation: transforming one string to another

$$s \rightarrow t$$

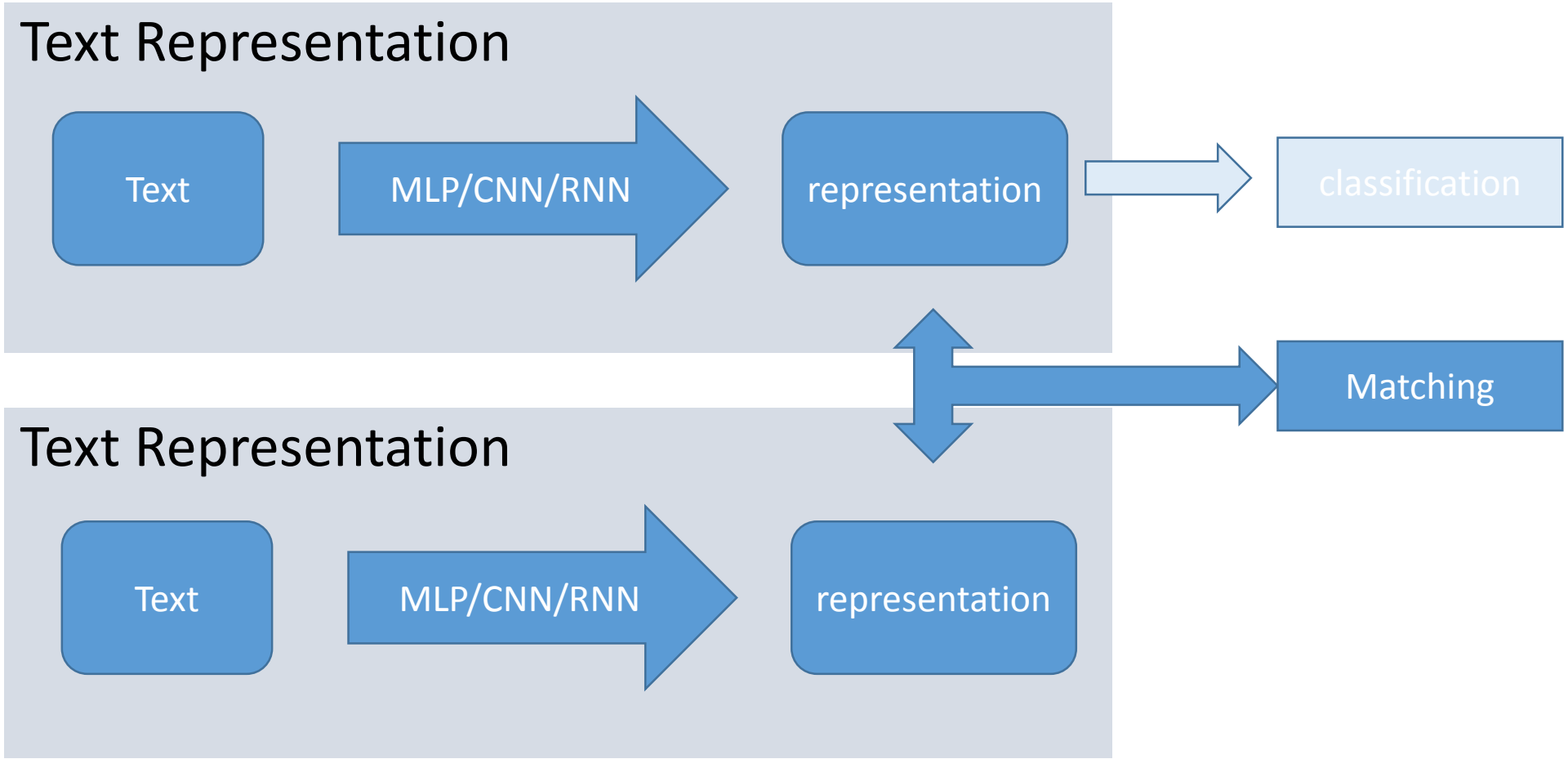
- Structured prediction: mapping string to structure

$$s \rightarrow s'$$

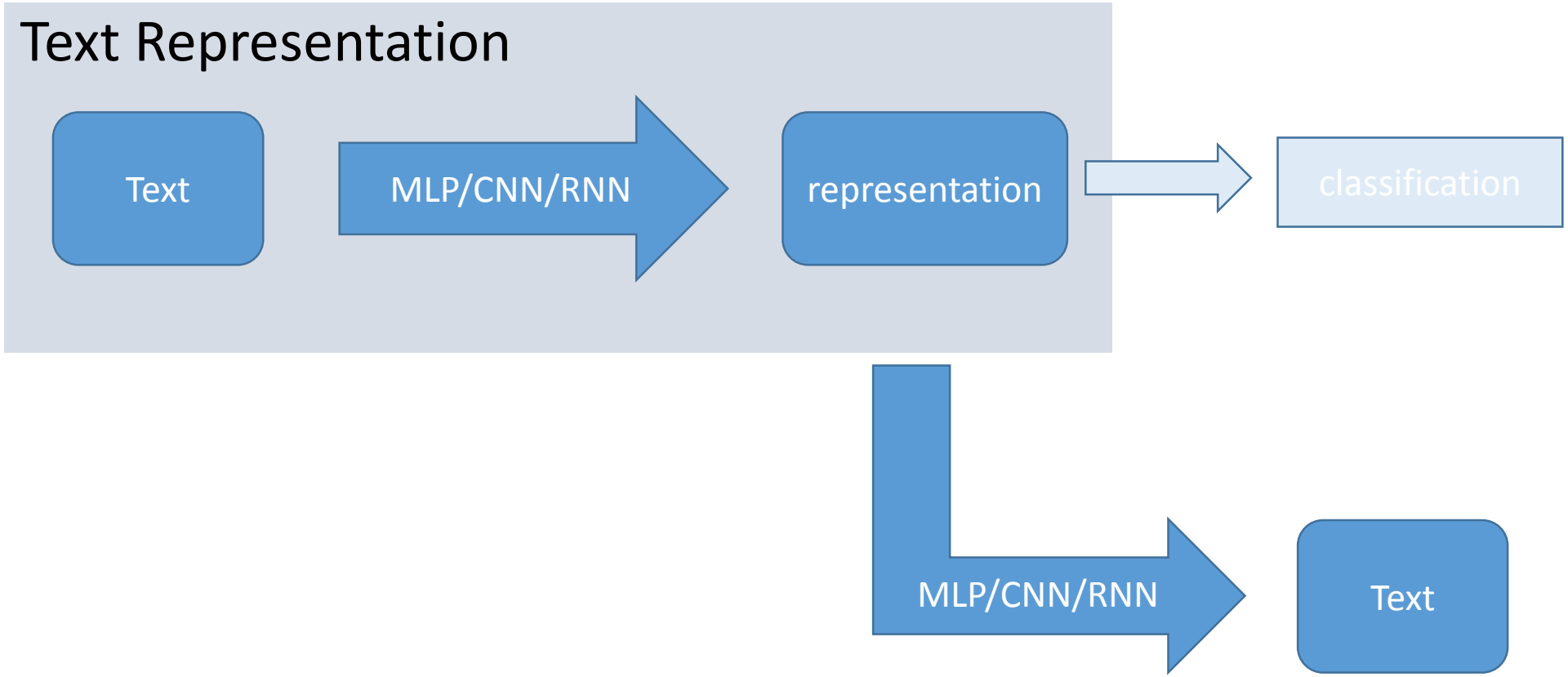
Why text classification?



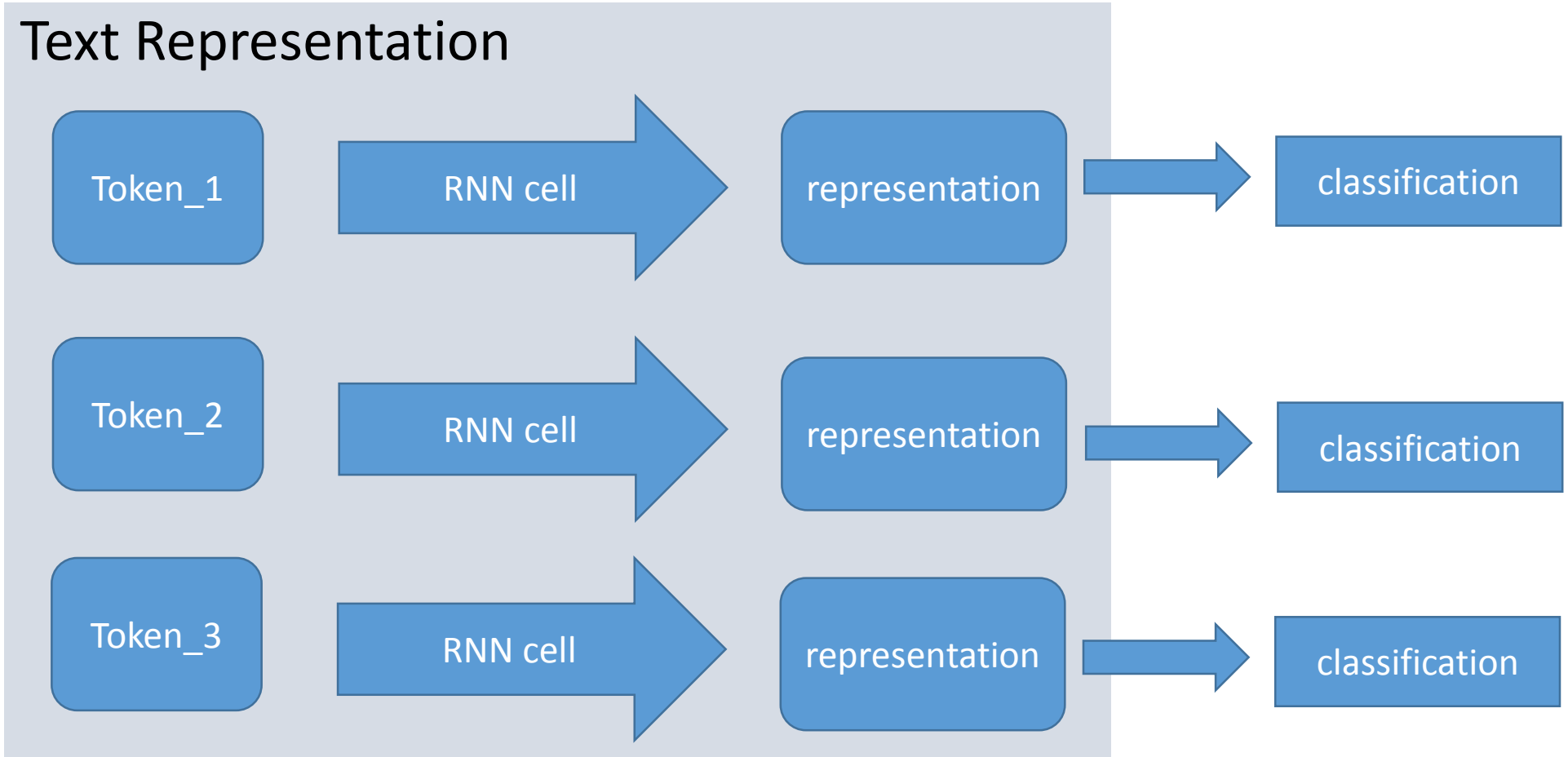
Why text classification?



Why text classification?



Why text classification?



Examples for LSTM

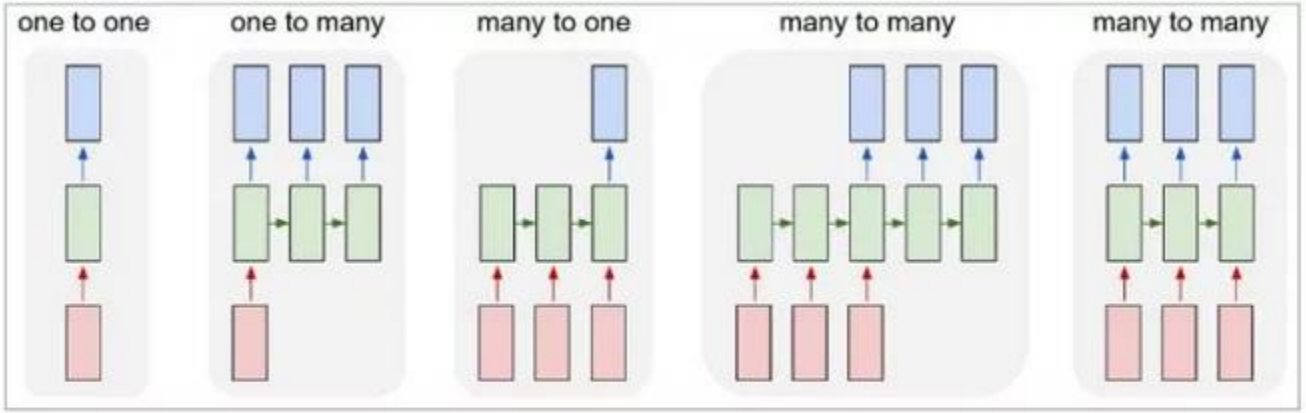


图1是普通的单个神经网络，图2是把单一输入转化为序列输出。 图3是把序列输入转化为单个输出 图4是把序列转化为序列，也就是 seq2seq 的做法。 图5是无时差的序列到序列转化，可以作为普通得语言模型

https://mp.weixin.qq.com/s/MhRrVW44dDX-PpWNqCWCOw

Fundamental Demo In Code with PyTorch pseudo code

- Model = LSTM/CNN/Capsule/…
- text,lable = Dataset.nextBatch()
- representation = Model(text)

- Classification = FC(representation) FC : Mapping to label size

- Translation = Decode(representation)

- Matching = Cosine(representation1, representation2)

- Sequential_labelling = FCs(representation**s**)

Contents

- Brief Introduction of TextZoo
- Why text classification？
- General Overview of Text Classification
- Overview of Text Classification in Neural Network approach.
- Architecture of TextZoo
- Conclusions

Overview

- Traditional Models
  - Naïve Bayes
  - SVM

- DL Models
  - ???CNN
  - ???RNN
  - ???NN

Traditional Classification

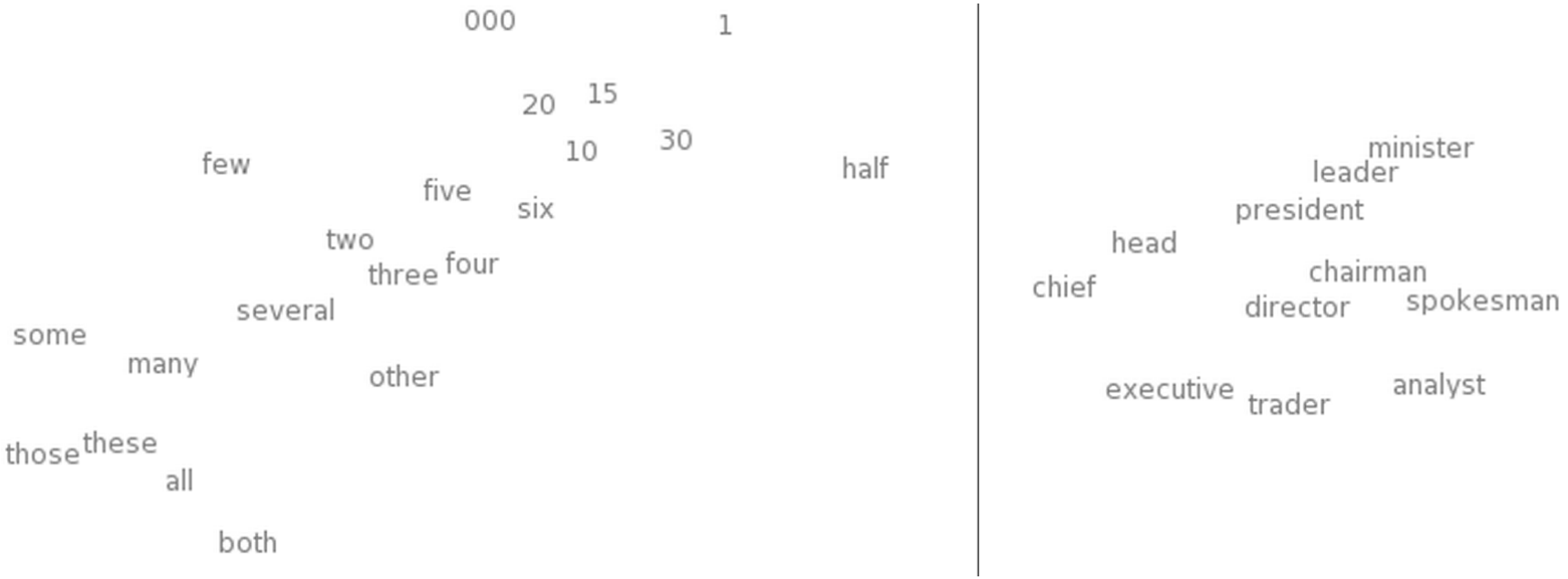
- SVM/Naïve Bayes
  - Bag-of-words(N-gram) hypothesis
  - Features :
    - TFIDF (unigram, N-gram)
    - POS, parser
    - hypernyms, WordNet
    - hand-coded rules
  - May needs “feature selection”
  - Good performance in long text

It performs better than you expected !!

Contents

- Brief Introduction of TextZoo
- Why text classification？
- General Overview of Text Classification
- Overview of Text Classification in Neural Network approach.
- Architecture of TextZoo
- Conclusions

Embedding and further DL models



[Distributional hypothesis]

linguistic items with similar distributions have similar meanings

https://en.wikipedia.org/wiki/Distributional_semantics

Localist representation

Size color … unknown

- BMW [1, 0, 0, 0, 0] [.3, .7, .2, .1, .5]
- Audi [0, 0, 0, 1, 0] [.5, .3, .2, .1, .0]
- Benz [0, 0, 1, 0, 0] [.2, .0, .31, .03, .01]
- Polo [0, 0, 0, 1, 0] [.1, .1, .5, .5, 0.2]

http://www.cs.toronto.edu/~bonner/courses/2014s/csc321/lectures/lec5.pdf

Distributed representation

- BMW [1, 0, 0, 0, 0]
- Audi [0, 0, 0, 1, 0]
- Benz [0, 0, 1, 0, 0]
- Polo [0, 0, 0, 1, 0]

Size color … unknown

[.3, .7, .2, .1, .5]

[.5, .3, .2, .1, .0]

[.2, .0, .31, .03, .01]

[.1, .1, .5, .5, 0.2]

How to get Distributed representation

- Matrix Factorization
  - Word-word Matrix
  - Document-word Matrix
    - PLSA
    - LDA
- Sample-based Prediction
  - NNLM
  - C & W
  - Word2vec

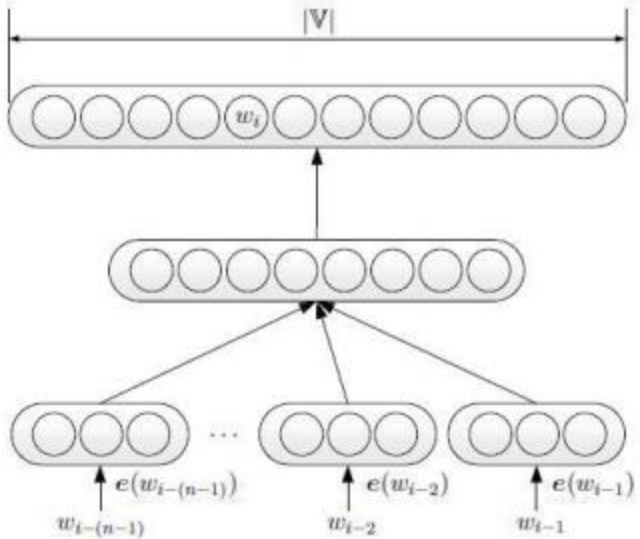
Glove is a combination between these two schools of approaches

Levy, Omer, and Yoav Goldberg. "Neural word embedding as implicit matrix factorization." *Advances in neural information processing systems*. 2014.

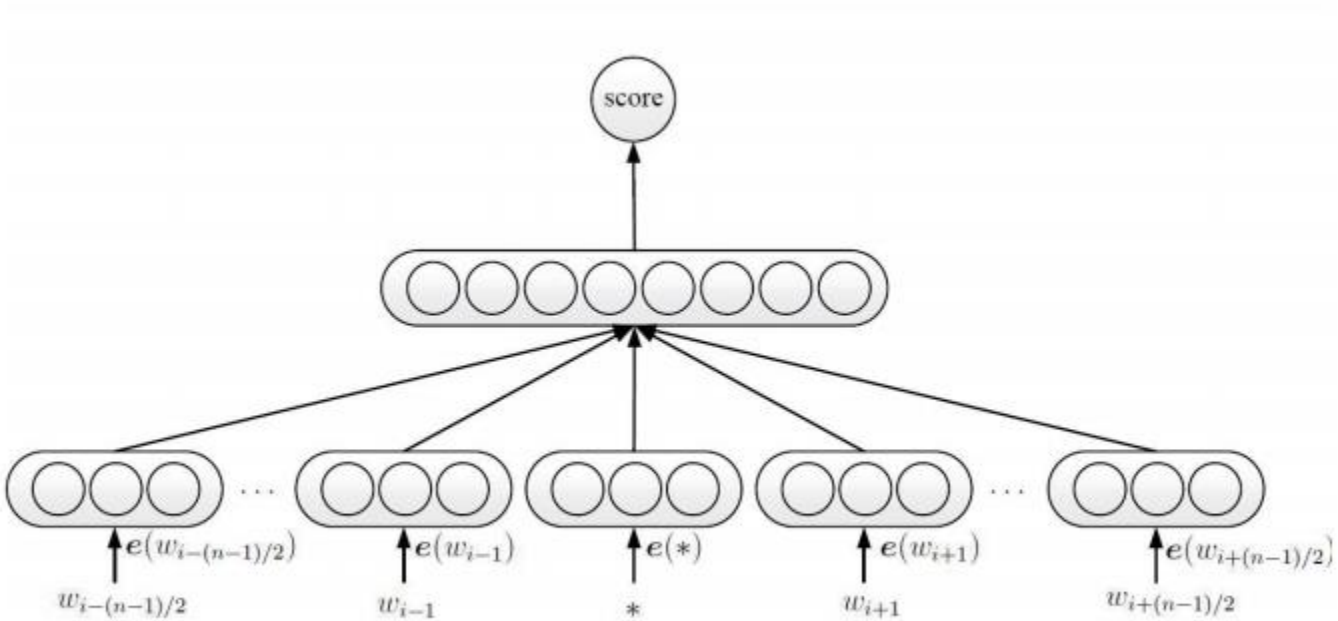
Why embedding is so hot?

- Only automatically build supervised pairs in unsupervised corpus
  - *Life is **complex**. It has both **real** and **imaginary** parts*

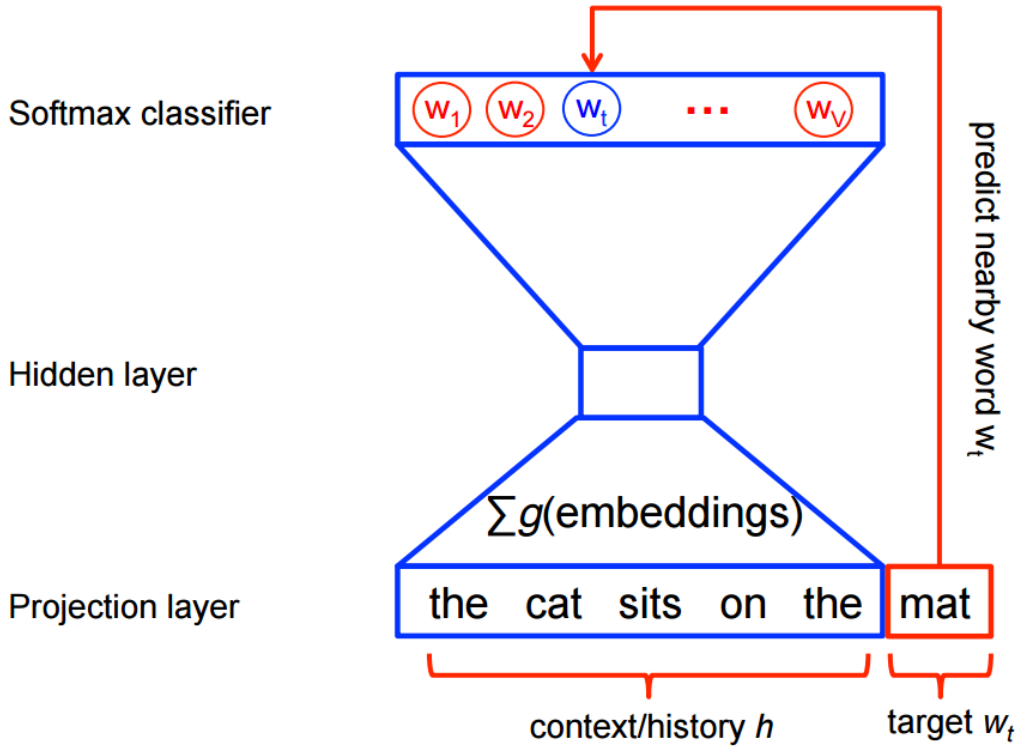
NNLM



C&W



Word2Vec


Softmax classifier
Hidden layer
Projection layer
w_1 w_2 w_t ... w_V
predict nearby word w_t
$\sum g$(embeddings)
the cat sits on the
mat
context/history h
target w_t

State-of-art Embedding

- Word2Vec
- Glove
- Many and many improved version of word embedding

  - Improved Word Representation Learning with Sememes
  - “Polysemy problem”
  - “Antonym problem”
  - Complex embedding [We are interested, now]
    - *life is complex, it has both real and imaginary parts*

Which is the most similar word of “Tencent” ?

May be “Baidu” or “pony” ?

Nie Jianyun said in SIGIR 2016 Chinese-Author Workshop, Tsinghua University, Beijing

Attention!!!

*Average Embedding may be a **problematic** practice for textual representation, especially in long text.*

Should add some supervised signals after embedding to reduce the noise !, like **Fastext**

Zhang, Xiang, Junbo Zhao, and Yann LeCun. "Character-level convolutional networks for text classification." *Advances in neural information processing systems*. 2015.

Embedding is everywhere!!!

- Word2vec
- Doc2vec
- Item2vec
- *Everything can be embed!!*

Embedding is a kind of approach, while **word vector** is a typical application of embedding

Wu, Ledell, et al. "StarSpace: Embed All The Things!." *arXiv preprint arXiv:1709.03856* (2017).

How to choose Word Vector

- Word2vec or Glove
  - *Depends on you final performance, not a prior test in linguistic/syntax regulation*

- **Embedding dim**, depends on scale of training dataset.
  - *Larger dataset, bigger dimension, but overfitting*.

- If possible, **train** the embedding on own your data.

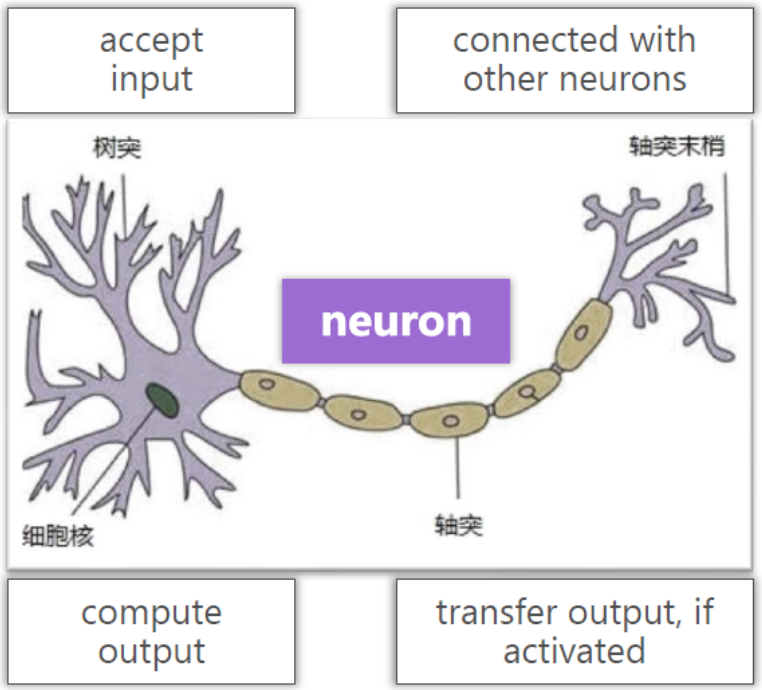
  Topic-relevant is somehow more important than the data size

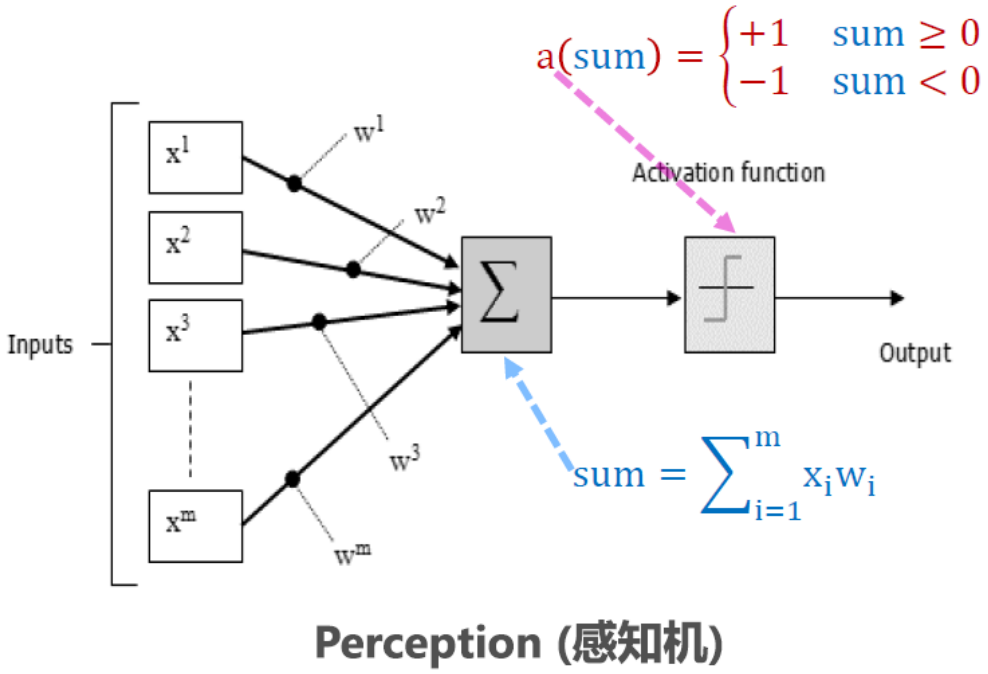
More features in DL

- POS Embedding
- CCG Embedding
- Extract matching Embedding
- Position Embedding

- Embed Every discrete features in Neural Network
  - If it is continuous, bucket it and make it discrete.

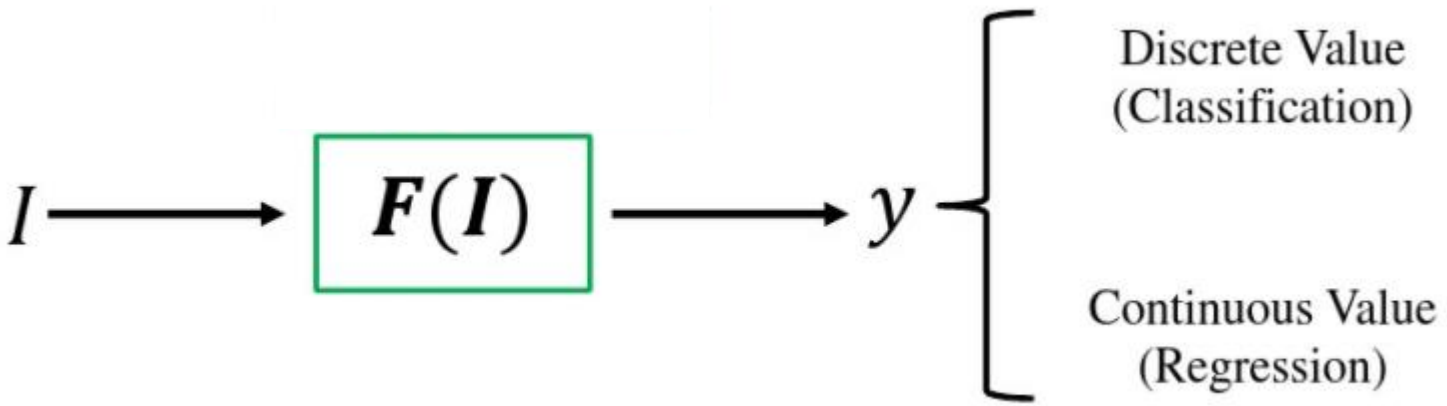
MLP





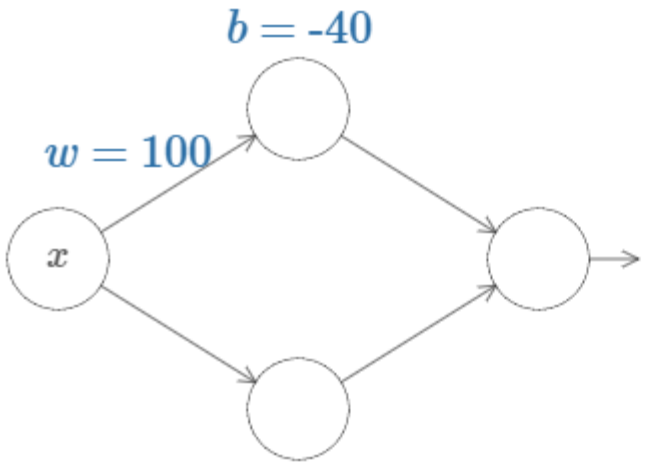
Perception (感知机)

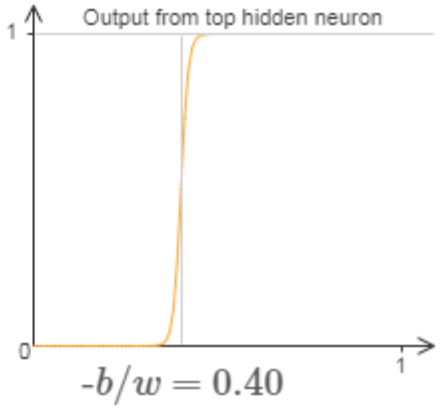
UAT in MLP



Multi-layer Non-linear Mapping - > **U**niversal **A**pproximation **T**heorem

A sample of θ(wx+b)



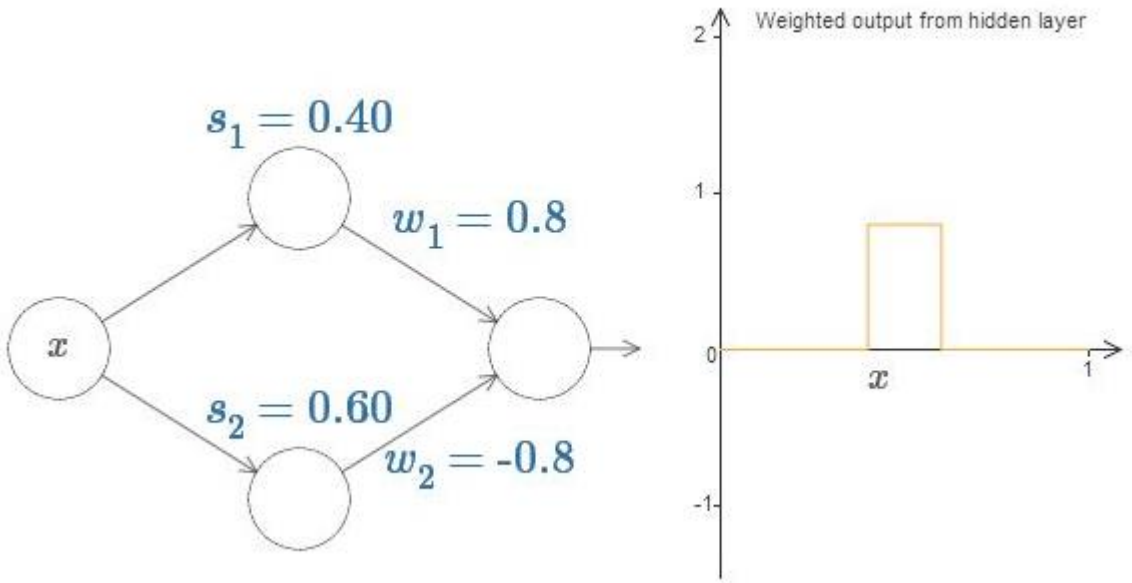


$$s = -b/w.$$

$$\sigma(wx + b), \text{where } \sigma(z) \equiv 1/(1 + e^{-z})$$

http://neuralnetworksanddeeplearning.com/chap4.html

An another sample

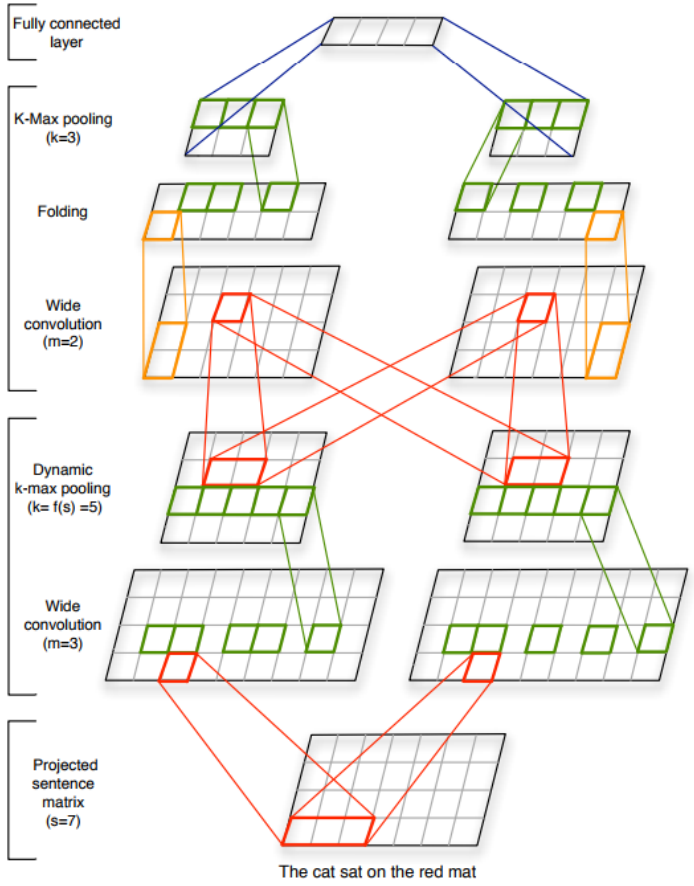


$$\sigma(wx + b), \text{where } \sigma(z) \equiv 1/(1 + e^{-z})$$

CNN

- Basic CNN
- Kalchbrenner N, Grefenstette E, Blunsom P. A convolutional neural network for modelling sentences[J]. arXiv preprint arXiv:1404.2188, 2014
- Kim CNN
- VDCNN

CNN [Kalchbrenner. et.al ACL 2014]



CNN [kim EMNLP 2014]

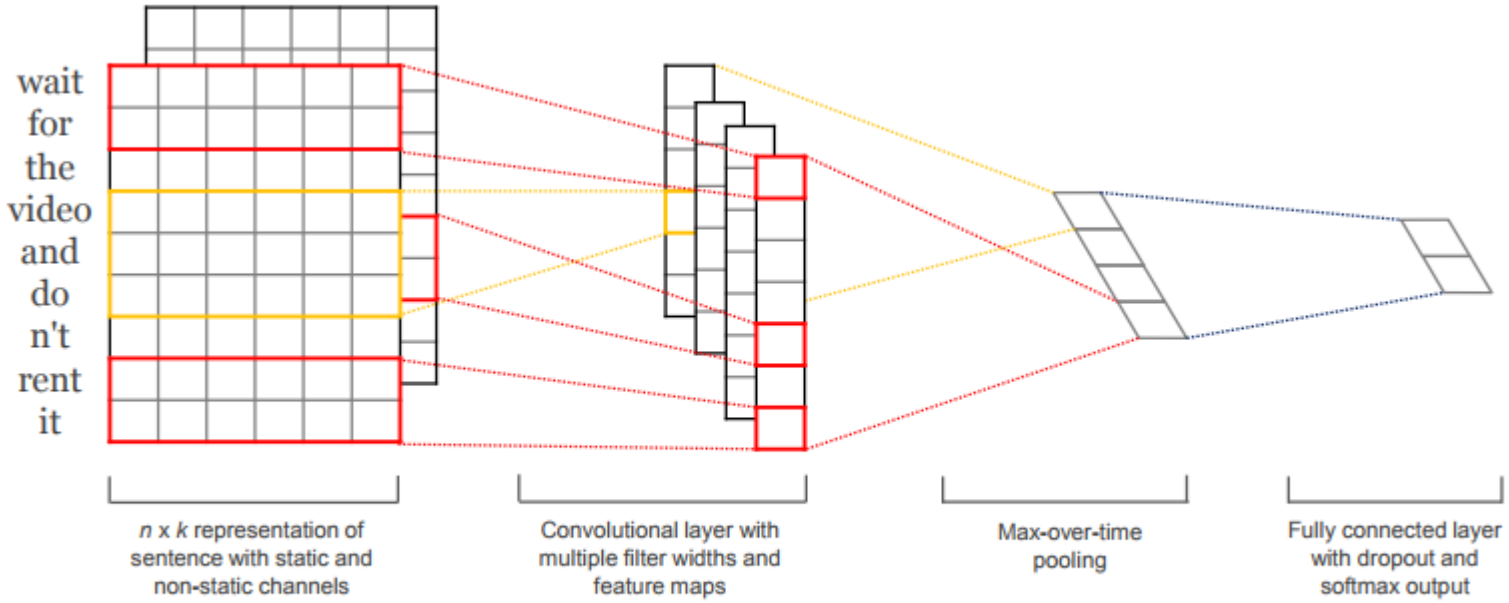


Figure 1: Model architecture with two channels for an example sentence.

Model	MR	SST-1	SST-2	Subj	TREC	CR	MPQA
CNN-rand	76.1	45.0	82.7	89.6	91.2	79.8	83.4
CNN-static	81.0	45.5	86.8	93.0	92.8	84.7	**89.6**
CNN-non-static	**81.5**	48.0	87.2	93.4	93.6	84.3	89.5
CNN-multichannel	81.1	47.4	**88.1**	93.2	92.2	**85.0**	89.4
RAE (Socher et al., 2011)	77.7	43.2	82.4	–	–	–	86.4
MV-RNN (Socher et al., 2012)	79.0	44.4	82.9	–	–	–	–
RNTN (Socher et al., 2013)	–	45.7	85.4	–	–	–	–
DCNN (Kalchbrenner et al., 2014)	–	48.5	86.8	–	93.0	–	–
Paragraph-Vec (Le and Mikolov, 2014)	–	**48.7**	87.8	–	–	–	–
CCAE (Hermann and Blunsom, 2013)	77.8	–	–	–	–	–	87.2
Sent-Parser (Dong et al., 2014)	79.5	–	–	–	–	–	86.3
NBSVM (Wang and Manning, 2012)	79.4	–	–	93.2	–	81.8	86.3
MNB (Wang and Manning, 2012)	79.0	–	–	**93.6**	–	80.0	86.3
G-Dropout (Wang and Manning, 2013)	79.0	–	–	93.4	–	82.1	86.1
F-Dropout (Wang and Manning, 2013)	79.1	–	–	**93.6**	–	81.9	86.3
Tree-CRF (Nakagawa et al., 2010)	77.3	–	–	–	–	81.4	86.1
CRF-PR (Yang and Cardie, 2014)	–	–	–	–	–	82.7	–
SVM_S (Silva et al., 2011)	–	–	–	–	**95.0**	–	–

FASTEX [EACL 2017]

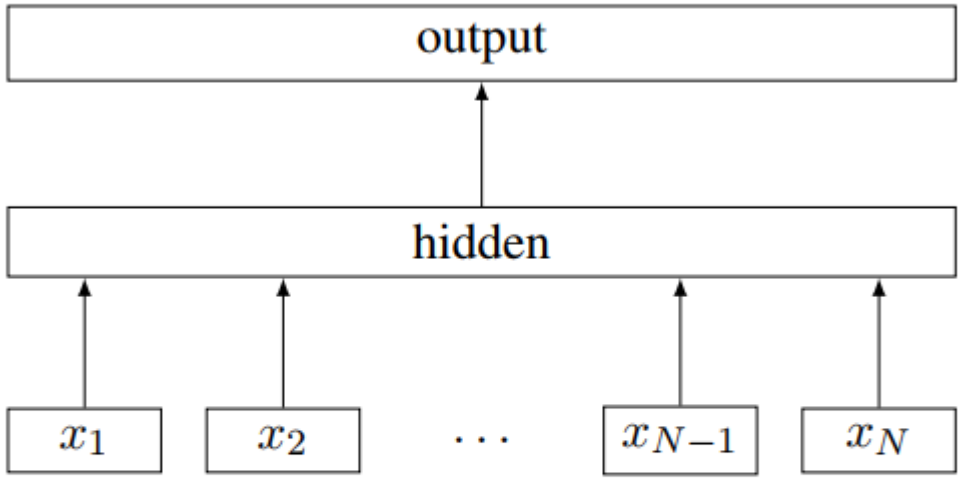


Figure 1: Model architecture of `fastText` for a sentence with N ngram features $x_1, \ldots, x_N$. The features are embedded and averaged to form the hidden variable.

Model	Yelp'13	Yelp'14	Yelp'15	IMDB
SVM+TF	59.8	61.8	62.4	40.5
CNN	59.7	61.0	61.5	37.5
Conv-GRNN	63.7	65.5	66.0	42.5
LSTM-GRNN	65.1	67.1	67.6	45.3
`fastText`	64.2	66.2	66.6	45.2

Why Mr. Lace chooses FasText

- Fast
- Input may a set of **keywords** instead of a sequential of words
  - (Group name)
- Label may be **inaccurate**

- Build more hand-code features would get comparable results

Very Large CNN [Conneau EACL]

Corpus:	AG	Sogou	DBP.	Yelp P.	Yelp F.	Yah. A.	Amz. F.	Amz. P.
Method	n-TFIDF	n-TFIDF	n-TFIDF	ngrams	Conv	Conv+RNN	Conv	Conv
Author	[Zhang]	[Zhang]	[Zhang]	[Zhang]	[Zhang]	[Xiao]	[Zhang]	[Zhang]
Error	7.64	2.81	1.31	4.36	37.95*	28.26	40.43*	4.93*
[Yang]	-	-	-	-	-	24.2	36.4	-

Table 4: Best published results from previous work. Zhang et al. (2015) best results use a Thesaurus data augmentation technique (marked with an *). Yang et al. (2016)'s hierarchical methods is particularly adapted to datasets whose samples contain multiple sentences.

Depth	Pooling	AG	Sogou	DBP.	Yelp P.	Yelp F.	Yah. A.	Amz. F.	Amz. P.
9	Convolution	10.17	4.22	1.64	5.01	37.63	28.10	38.52	4.94
9	KMaxPooling	9.83	3.58	1.56	5.27	38.04	28.24	39.19	5.69
9	MaxPooling	9.17	3.70	1.35	4.88	36.73	27.60	37.95	4.70
17	Convolution	9.29	3.94	1.42	4.96	36.10	27.35	37.50	4.53
17	KMaxPooling	9.39	3.51	1.61	5.05	37.41	28.25	38.81	5.43
17	MaxPooling	8.88	3.54	1.40	4.50	36.07	27.51	37.39	4.41
29	Convolution	9.36	3.61	1.36	4.35	**35.28**	27.17	37.58	**4.28**
29	KMaxPooling	**8.67**	**3.18**	1.41	4.63	37.00	27.16	38.39	4.94
29	MaxPooling	8.73	3.36	**1.29**	**4.28**	35.74	**26.57**	**37.00**	4.31

Table 5: Testing error of our models on the 8 data sets. No data preprocessing or augmentation is used.

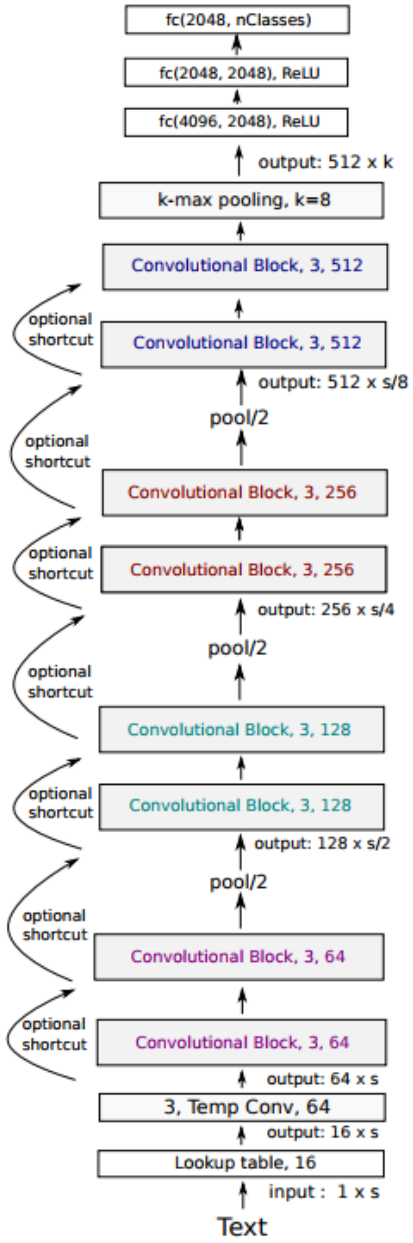


Figure 1: VDCNN architecture.

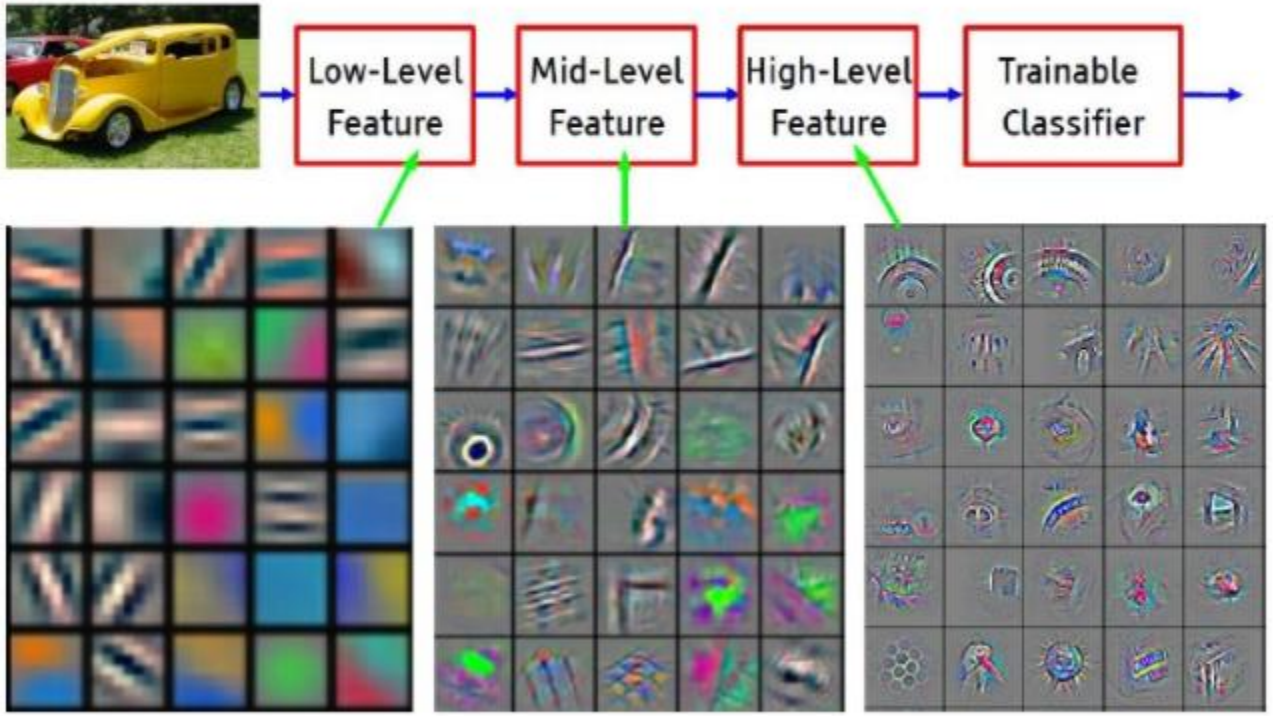
Go deeper or not?

- DEEP
  - Slower
  - Overfitting
    - More Parameters, more data need to feed
  - Hard for convergence
    - Highway network
    - Residual Block
    - Inception

- Shallow: one-layer
  - Fast
  - Less data, es. Fastext.

Go deeper or not?

- Image recognition: Pixel → edge → texton → motif → part → object
- Text: Character → word → word group → clause → sentence → story
- Speech: Sample → spectral band → sound → … → phone → phoneme → word



Feature visualization of convolutional net trained on ImageNet from [Zeiler & Fergus 2013]

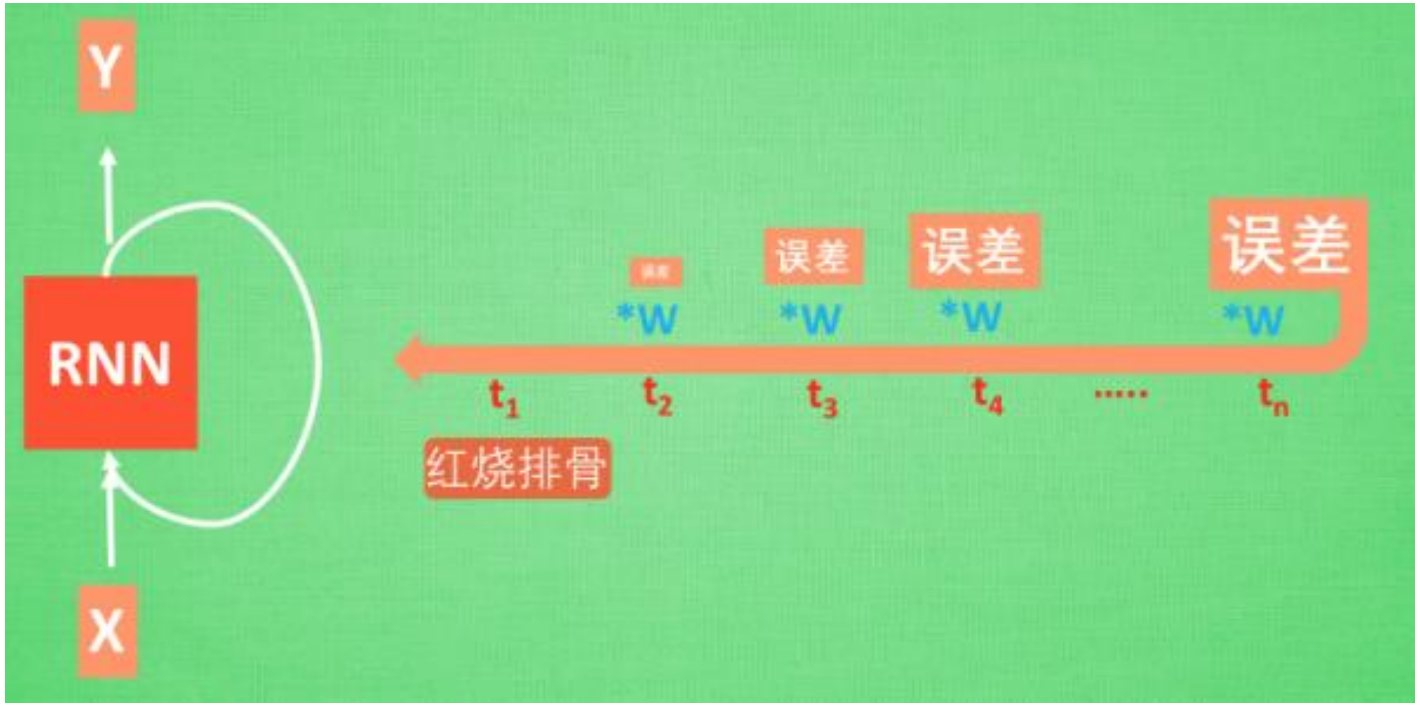
Modified from Prof. LeCun and Prof. Bengio

RNN and its Variant

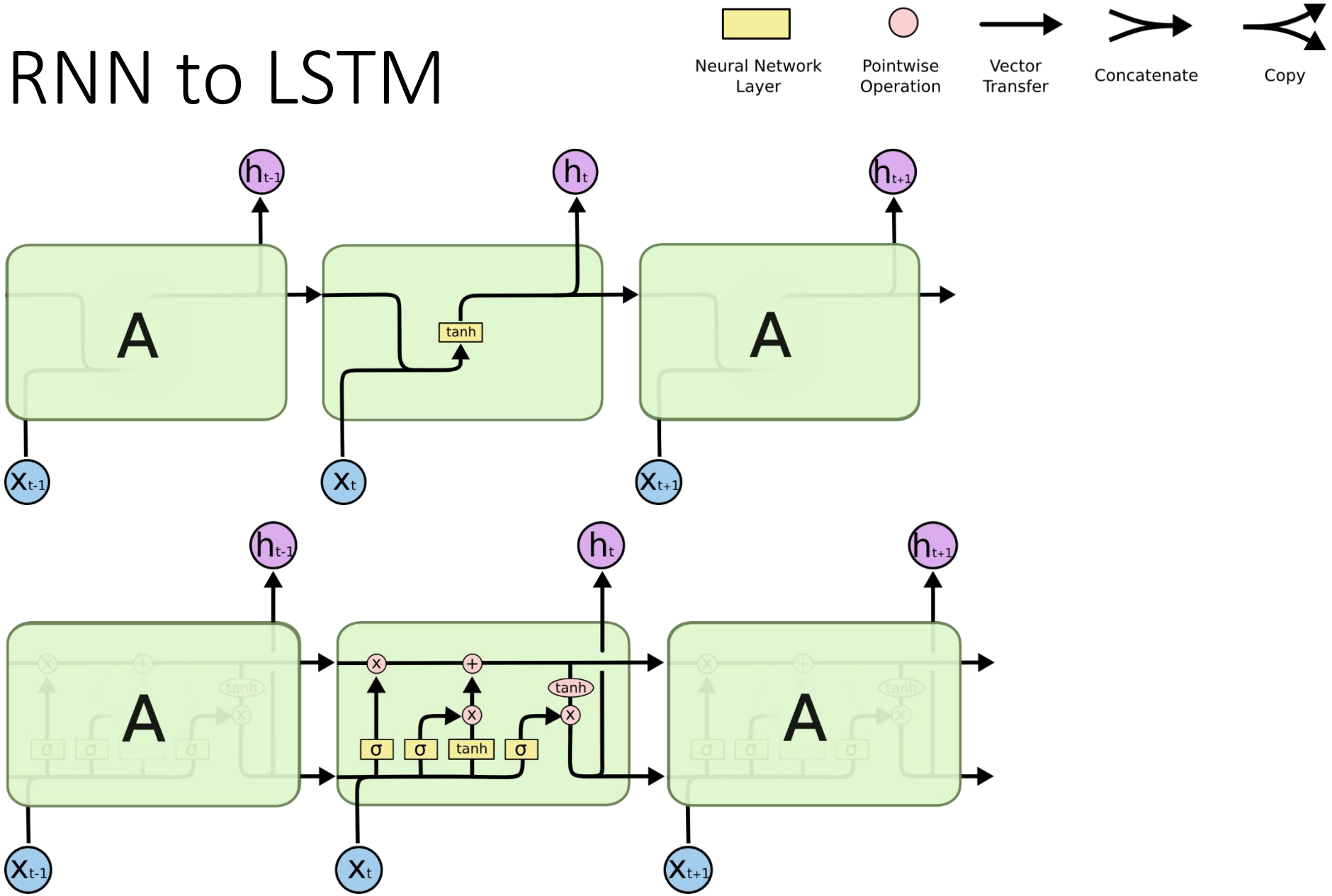
- RNN
- LSTM
- LSTM + mean
- LSTM + bidirectional
- LSTM + Attention
- LSTM + Stack
- LSTM + Self-Attention
- TreeLSTM

Bias in RNN

Bias in RNN

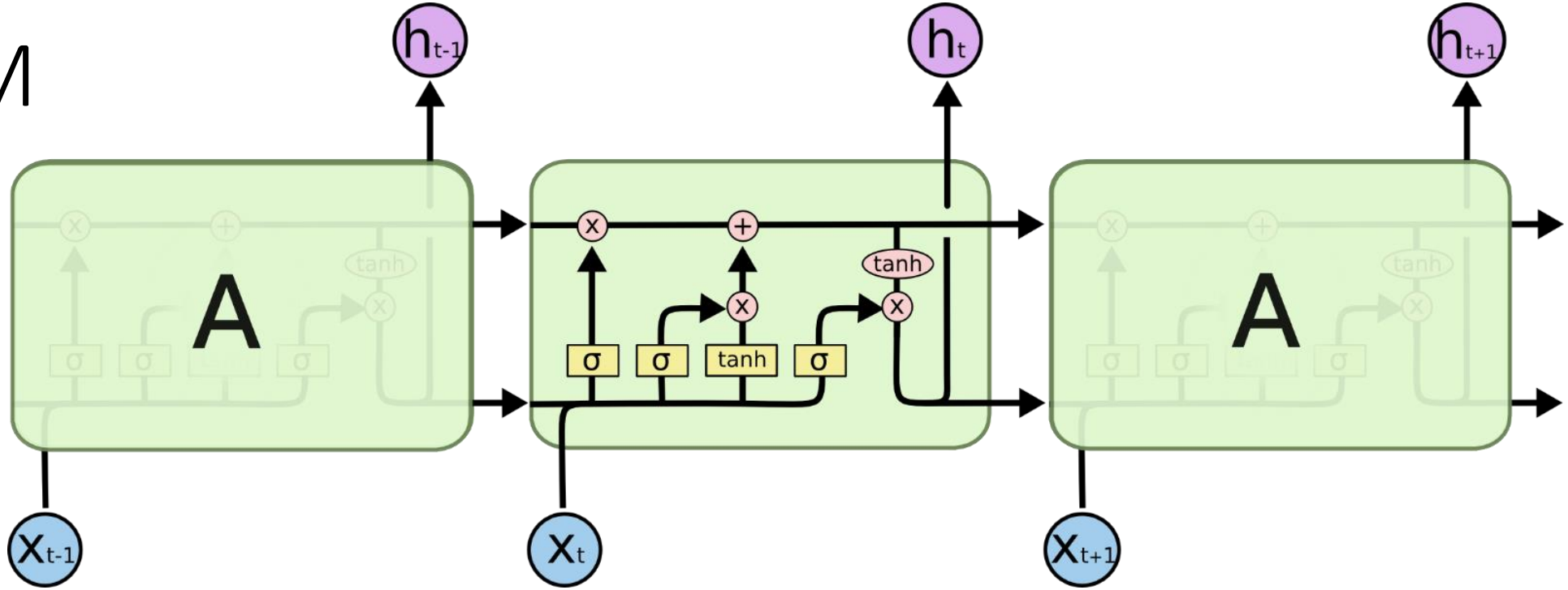


From RNN to LSTM



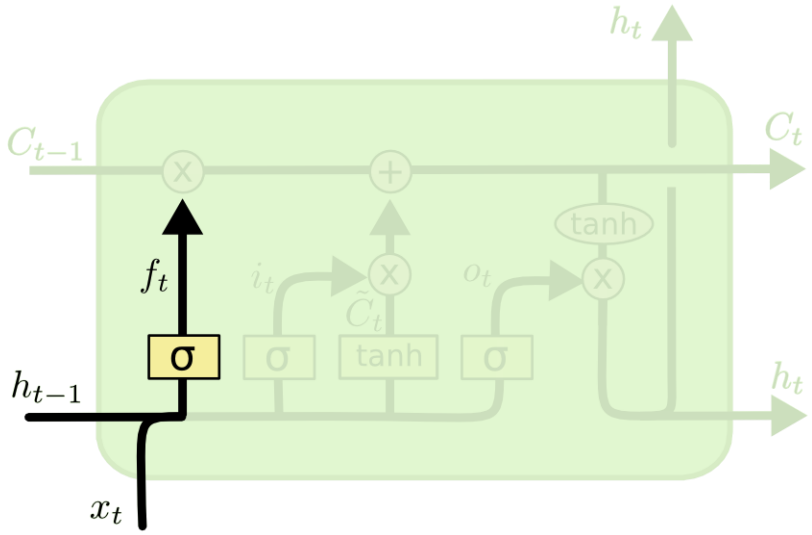
http://colah.github.io/posts/2015-08-Understanding-LSTMs/

LSTM



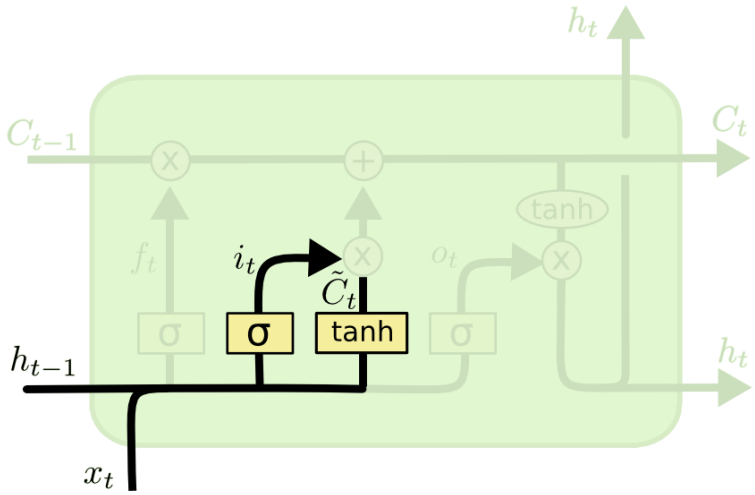
- How many gates ?
- Difference between cell and the hidden state?
- How many parameters in a LSTM?

Forget gate



$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

Input gate



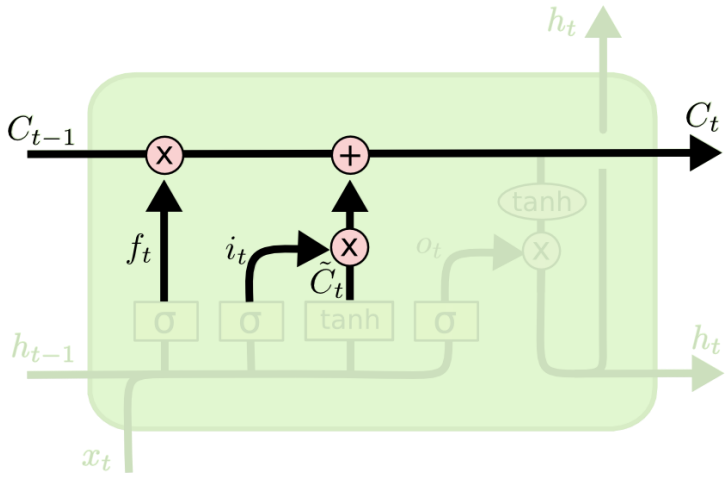
$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$

replace tanh with **softsign** (not softmax) activation for prevent **overfitting**

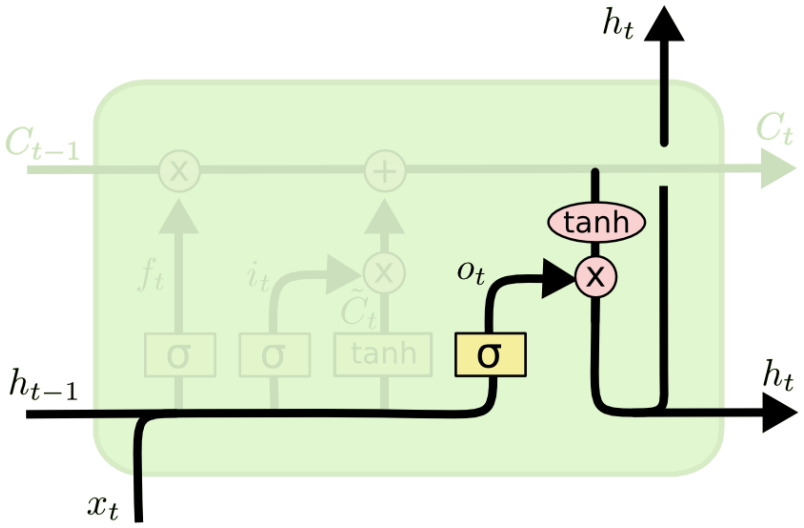
https://zhuanlan.zhihu.com/p/21952042

Forgotten + input



$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

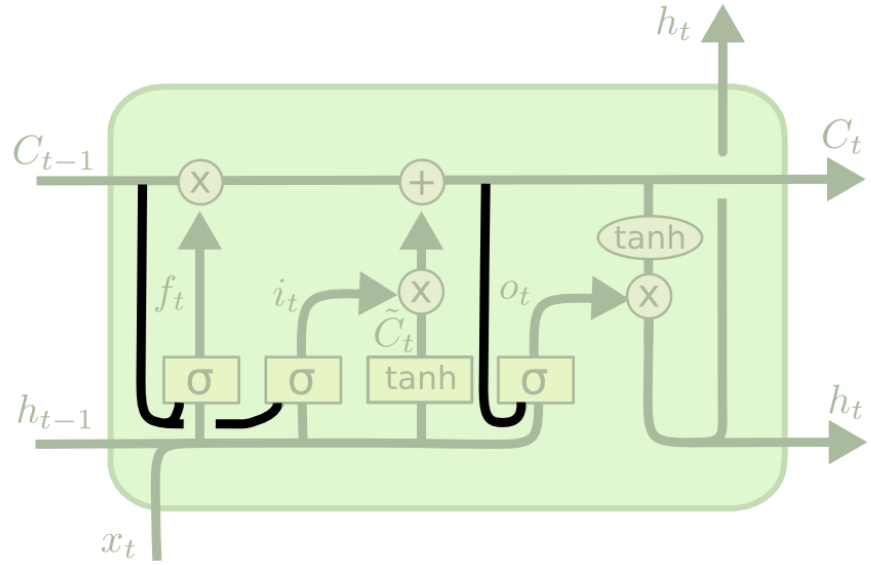
Output Gate



$$o_t = \sigma\left(W_o\left[h_{t-1}, x_t\right] + b_o\right)$$

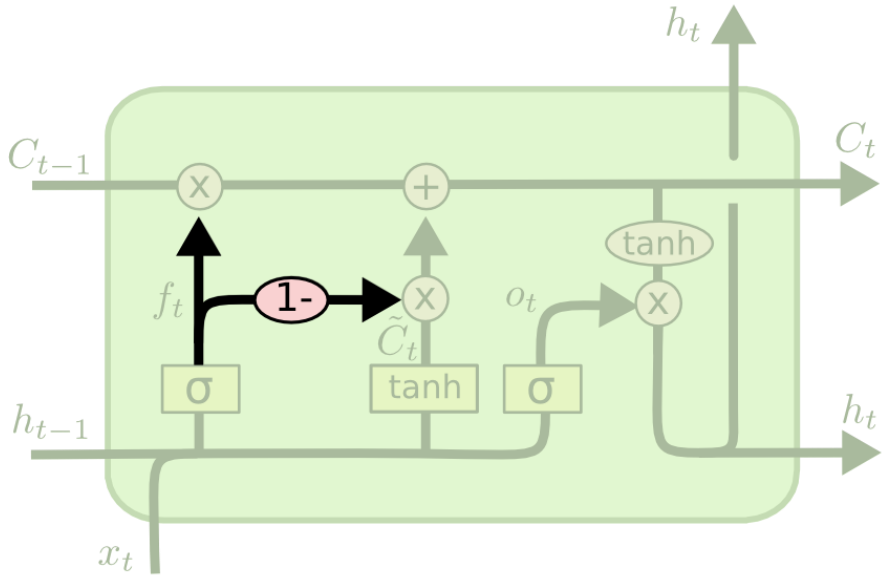
$$h_t = o_t * \tanh\left(C_t\right)$$

LSTM **Variants:** Peephole connections



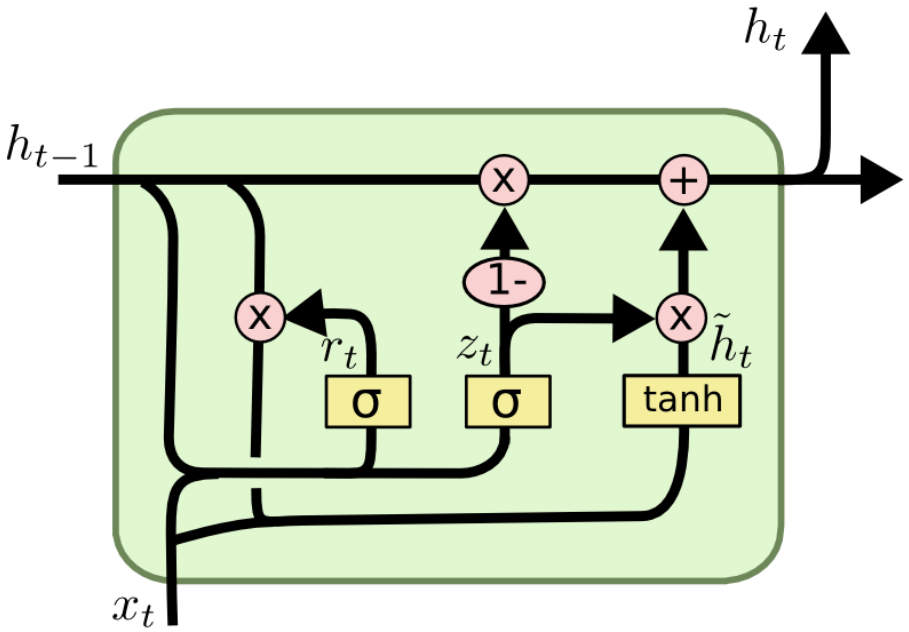
$$f_t = \sigma\left(W_f \cdot [\boldsymbol{C_{t-1}}, h_{t-1}, x_t] \; + \; b_f\right)$$
$$i_t = \sigma\left(W_i \cdot [\boldsymbol{C_{t-1}}, h_{t-1}, x_t] \; + \; b_i\right)$$
$$o_t = \sigma\left(W_o \cdot [\boldsymbol{C_t}, h_{t-1}, x_t] \; + \; b_o\right)$$

LSTM **Variants:** coupled forget and input gates



$$C_t = f_t * C_{t-1} + (1 - \boldsymbol{f_t}) * \tilde{C}_t$$

LSTM **Variants:** GRU



$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$

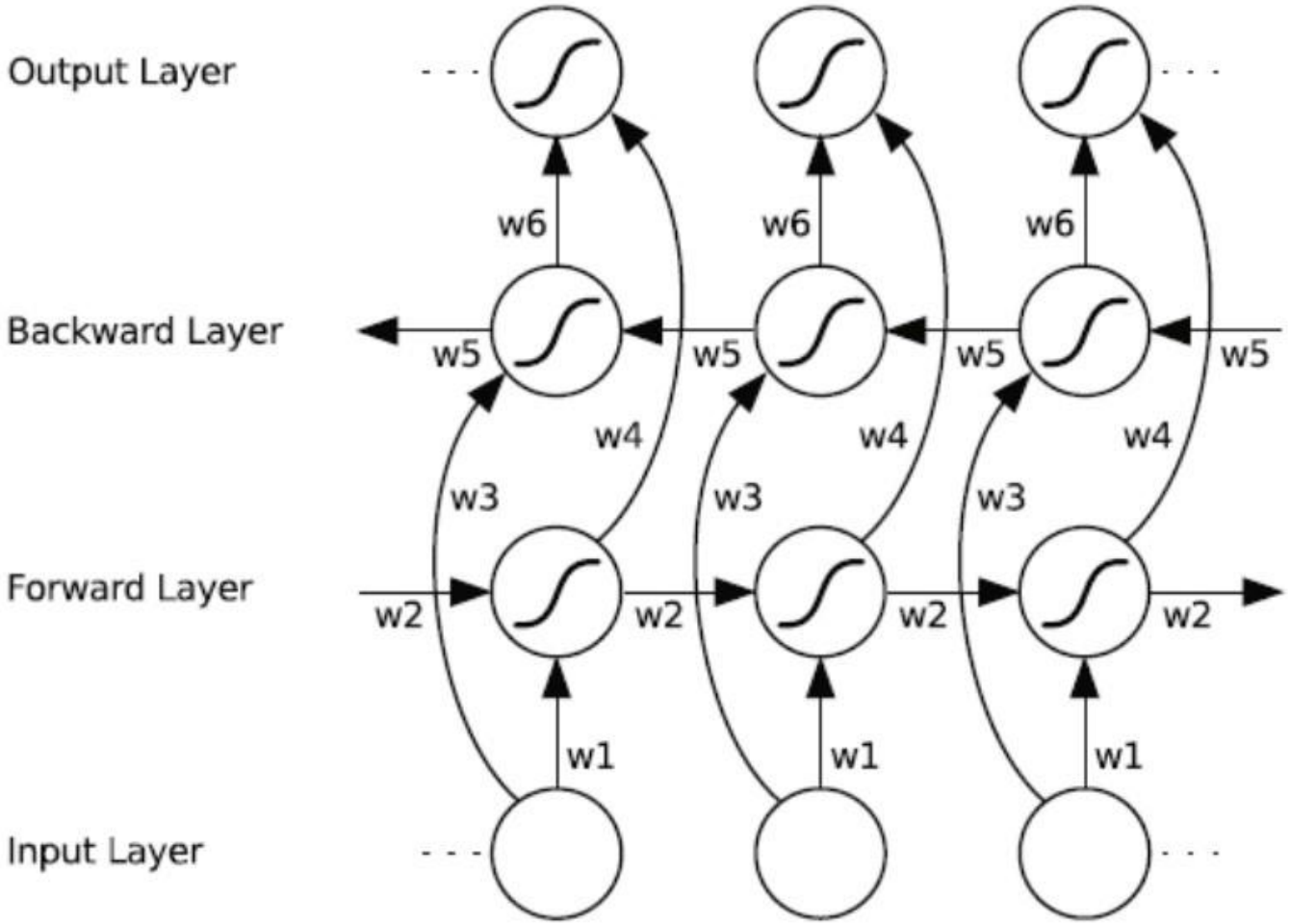
$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$

$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$

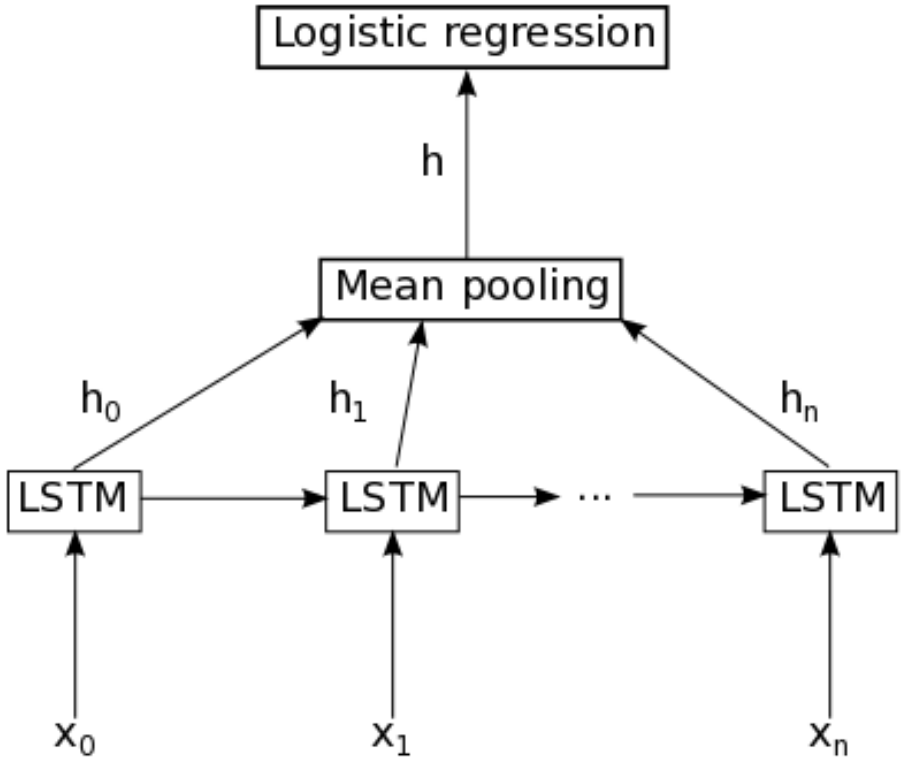
$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

- ✓ Hidden = Cell
- ✓ Forget gate + input gate =1

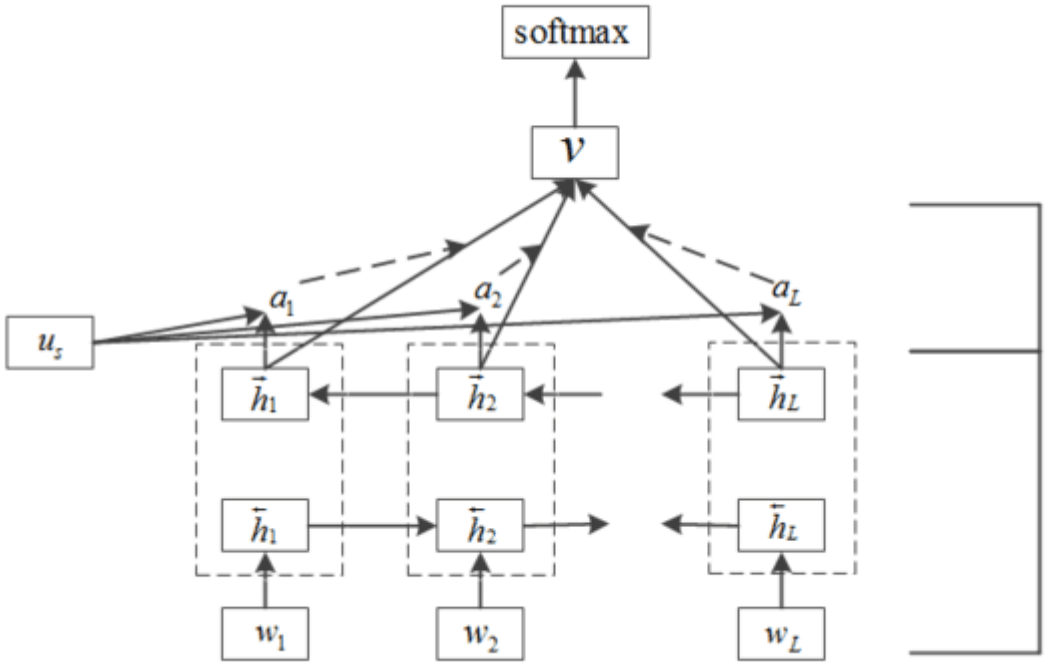
BiLSTM


Output Layer
Backward Layer
Forward Layer
Input Layer
w1
w2
w3
w4
w5
w6

Last or Mean?


Logistic regression
h
Mean pooling
h_0
h_1
h_n
LSTM
LSTM
...
LSTM
x_0
x_1
x_n

RNN/LSTM with Attention



BIGRU : 93%

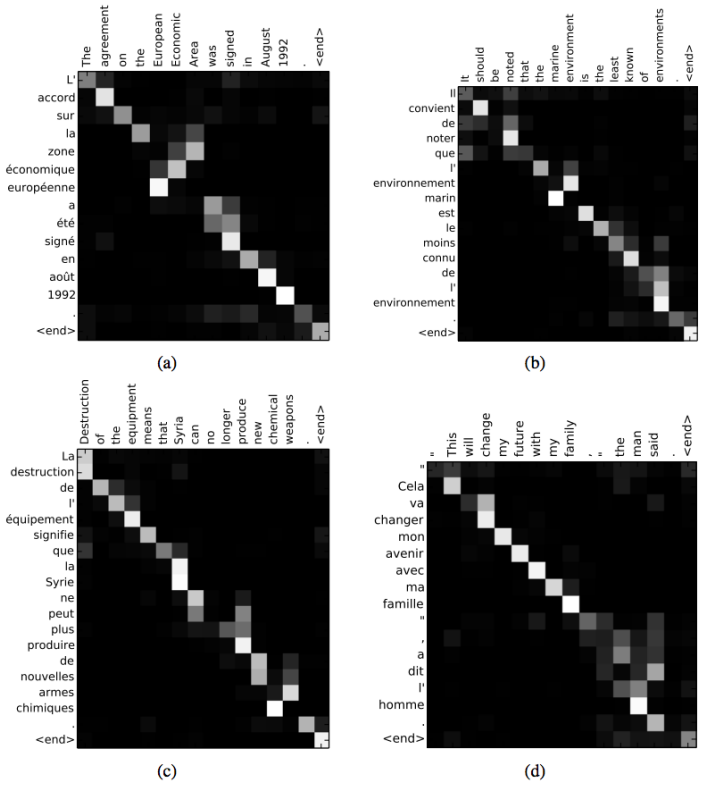
BILSTM : 91.43%

BIGRU_ATTENTION : 95.4%

BILSTM_ATTENTION : 96.2%

https://www.jianshu.com/p/4fbc4939509f

Visualization of Attention in RNN/LSTM



Machine Translation

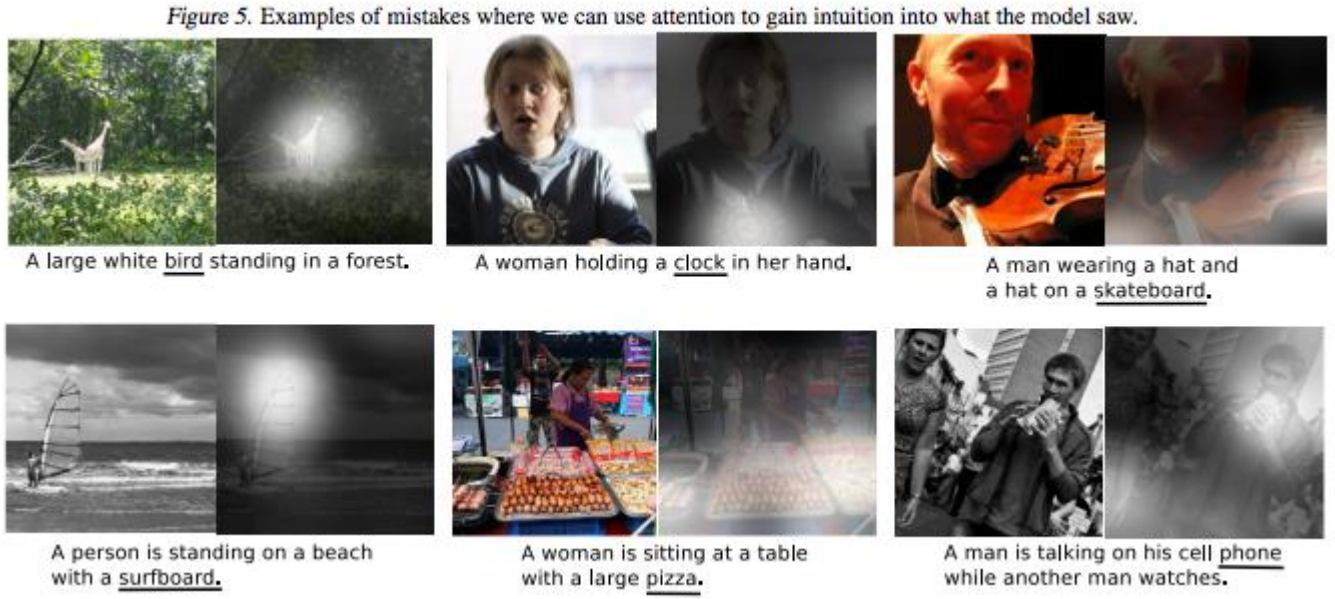
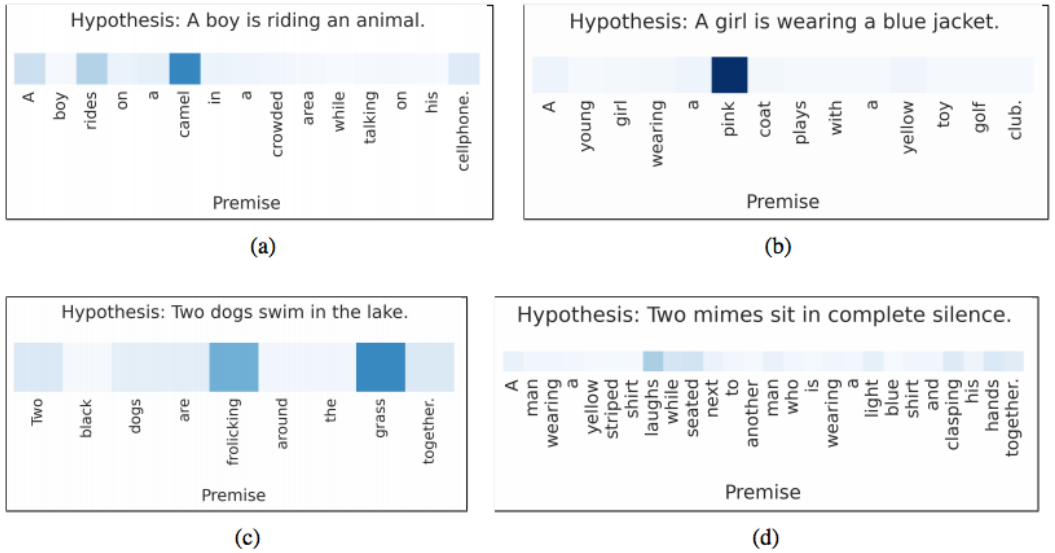
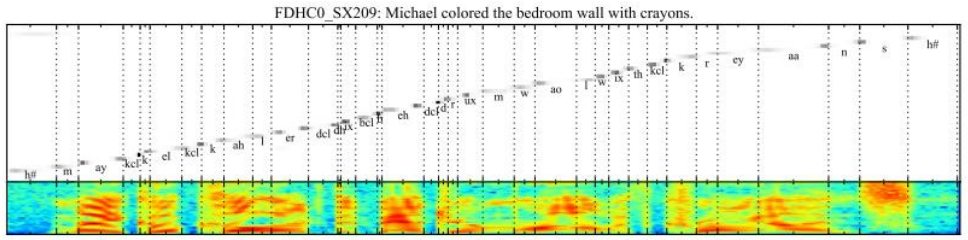


Image Caption

Visualization of Attention in RNN/LSTM



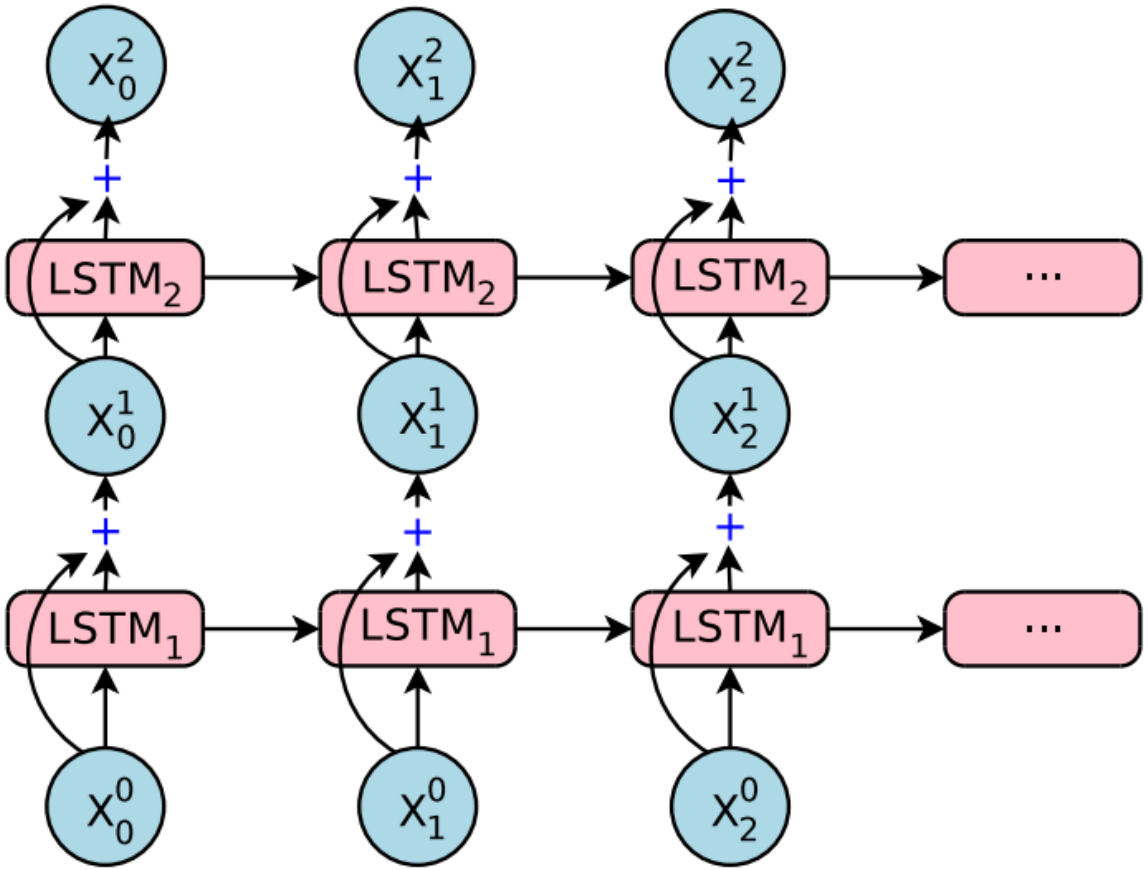
Sematic Entailment



Speech Recognition

Deeper LSTM

Deeper LSTM



Deep is not necessary, but more data!!!

CNN/RNN

- Comparative Study of CNN and RNN for Natural Language Processing

			performance	lr	hidden	batch	sentLen	filter_size	margin
TextC	SentiC (acc)	CNN	82.38	0.2	20	5	60	3	–
		GRU	**86.32**	0.1	30	50	60	–	–
		LSTM	84.51	0.2	20	40	60	–	–
	RC (F1)	CNN	68.02	0.12	70	10	20	3	–
		GRU	**68.56**	0.12	80	100	20	–	–
		LSTM	66.45	0.1	80	20	20	–	–
SemMatch	TE (acc)	CNN	77.13	0.1	70	50	50	3	–
		GRU	**78.78**	0.1	50	80	65	–	–
		LSTM	77.85	0.1	80	50	50	–	–
	AS (MAP & MRR)	CNN	(**63.69,65.01**)	0.01	30	60	40	3	0.3
		GRU	(62.58,63.59)	0.1	80	150	40	–	0.3
		LSTM	(62.00,63.26)	0.1	60	150	45	–	0.1
	QRM (acc)	CNN	**71.50**	0.125	400	50	17	5	0.01
		GRU	69.80	1.0	400	50	17	-	0.01
		LSTM	71.44	1.0	200	50	17	-	0.01
SeqOrder	PQA (hit@10)	CNN	54.42	0.01	250	50	5	3	0.4
		GRU	**55.67**	0.1	250	50	5	–	0.3
		LSTM	55.39	0.1	300	50	5	–	0.3
ContextDep	POS tagging (acc)	CNN	94.18	0.1	100	10	60	5	–
		GRU	93.15	0.1	50	50	60	–	–
		LSTM	93.18	0.1	200	70	60	–	–
		Bi-GRU	94.26	0.1	50	50	60	–	–
		Bi-LSTM	**94.35**	0.1	150	5	60	–	–

RNN vs CNN

- RNN
  - 序列结构
  - 强调高阶关系
  - 位置跳跃的依赖
- 速度
  - 更慢，串行
  - 方便定长，通过attention

- CNN
  - 两个句子关系
  - N-gram匹配更重要的match 场景
  - 局部依赖关系
- 速度
  - 可以并行，更灵活
  - 输出不定长，跟文本长度有关

CNN vs RNN vs their Hybrids

Neural Network Model	Avg. Accuracy
Feed-Forward (Word Embeddings) [1]	58.4%
Feed-Forward (Feature Vectors) [1]	66.8%
CNN	66.7%
LSTM	72.5%
CNN-LSTM	69.7%
LSTM-CNN	75.2%

http://blog.csdn.net/youngair/article/details/78013352

Dimensional Sentiment Analysis Using a Regional CNN-LSTM Model

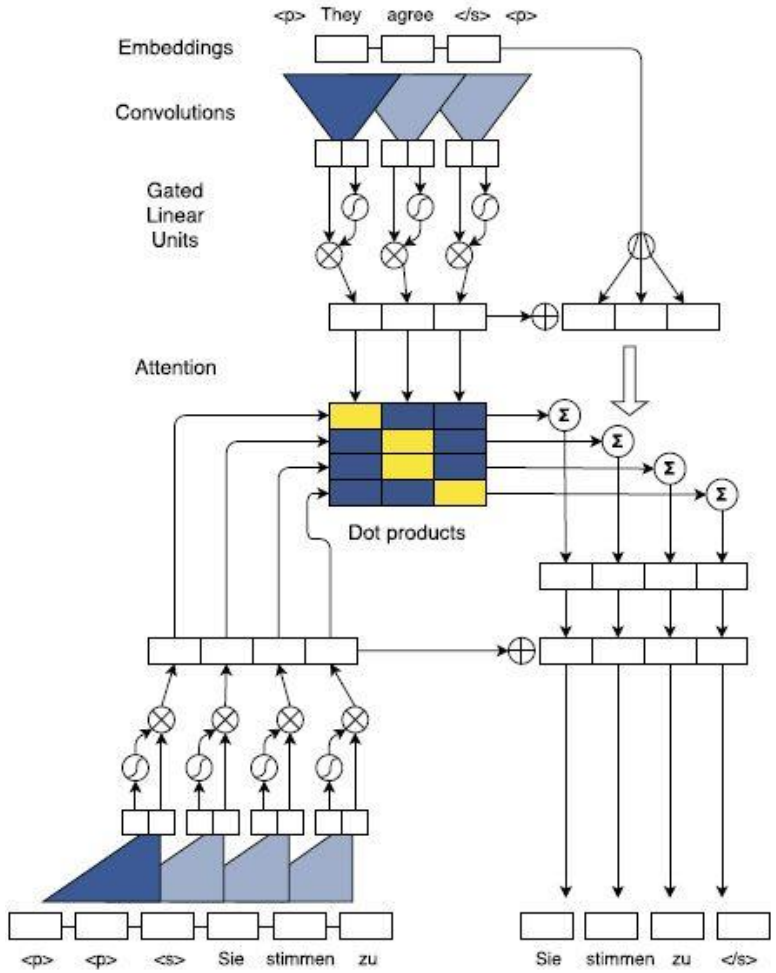
From a Industrial perspective

- Add **features**.
- **Understanding** your data **:** pay more attention on data **preparation.**
- **Parameter** adjusting with a robust setting
  - *Oh, overfit*
- **Model** is not very important, especially data is not low-quality.
  - *Models differs slightly in low-quality data.*
- Trade-off between performance and **efficiency**
  - *For example, multi-size kennels is better but slower!*

Related Models

- Do not directly aims at this task, but also aims to build a text representation.
  - ConvS2S
  - Attention is all you need
  - Dynamic Memory Network

Conv S2S


<p> They agree </s> <p>
Embeddings
Convolutions
Gated Linear Units
Attention
Dot products
<p> <p> <s> Sie stimmen zu
Sie stimmen zu </s>

Attention is all you need

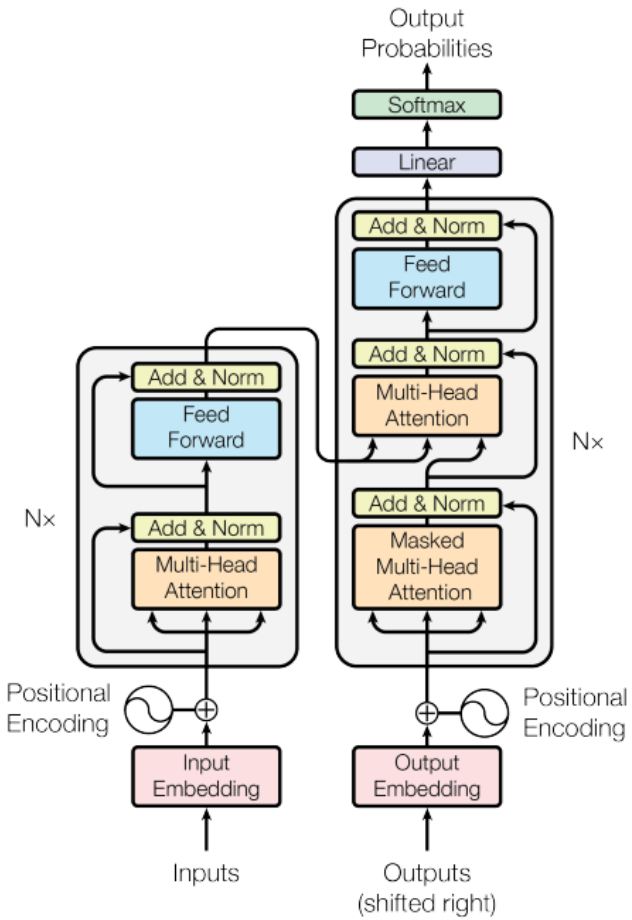
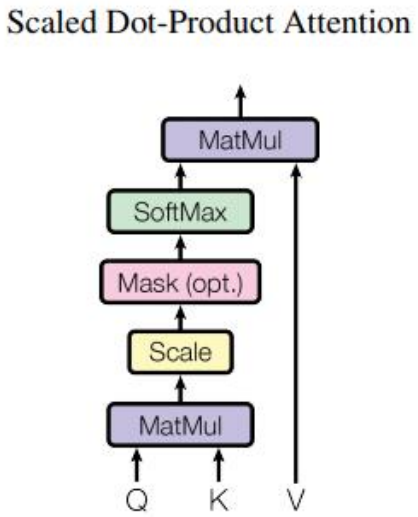
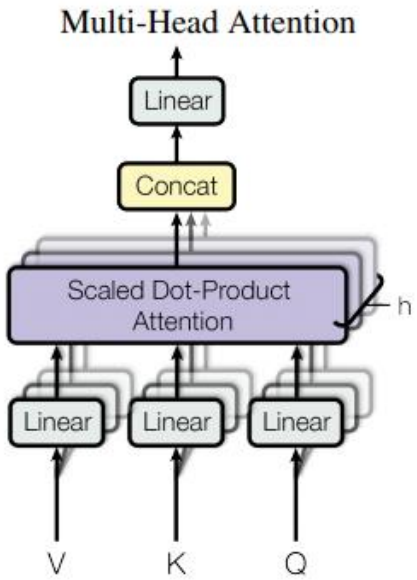


Figure 1: The Transformer - model architecture.





Self-Attention

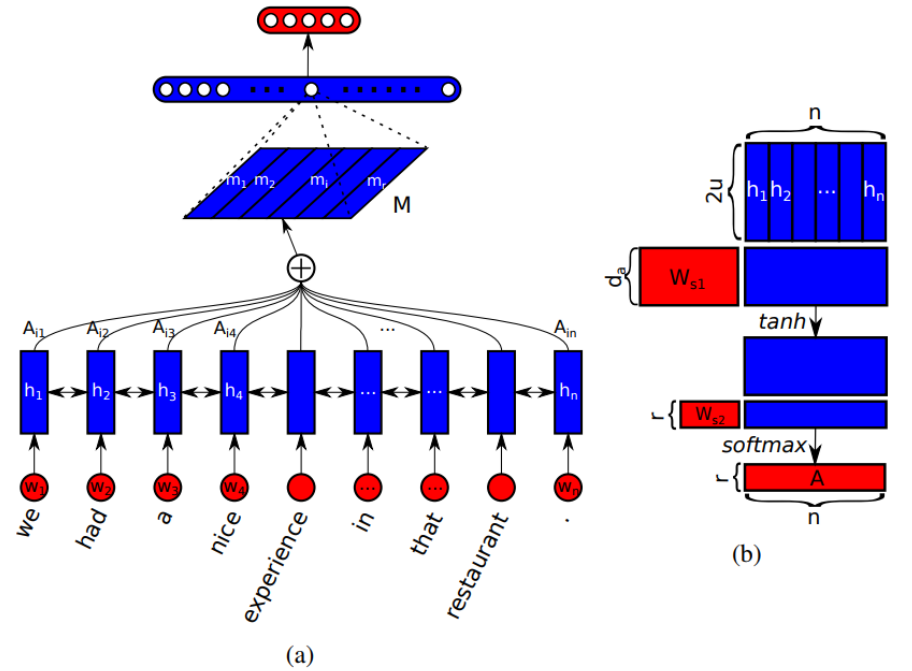
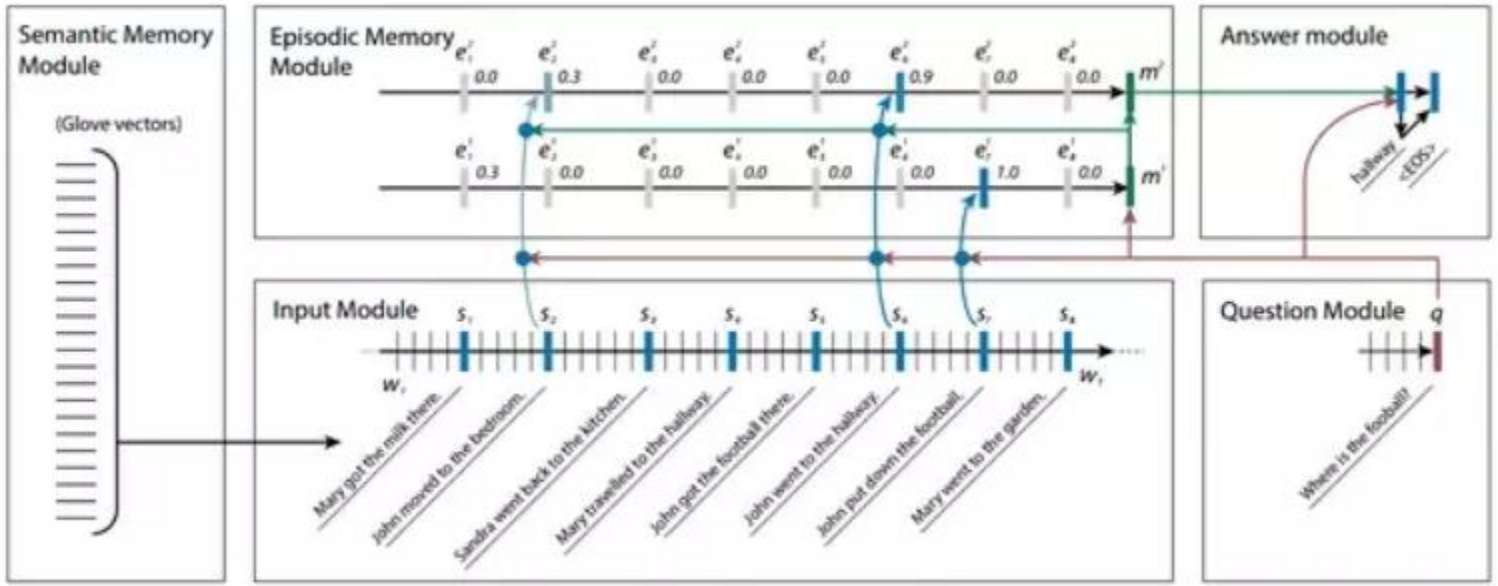


Figure 1: A sample model structure showing the sentence embedding model combined with a fully connected and softmax layer for sentiment analysis (a). The sentence embedding M is computed as multiple weighted sums of hidden states from a bidirectional LSTM ($\mathbf{h_1}, ..., \mathbf{h_n}$), where the summation weights ($A_{i1}, ..., A_{in}$) are computed in a way illustrated in (b). Blue colored shapes stand for hidden representations, and red colored shapes stand for weights, annotations, or input/output.

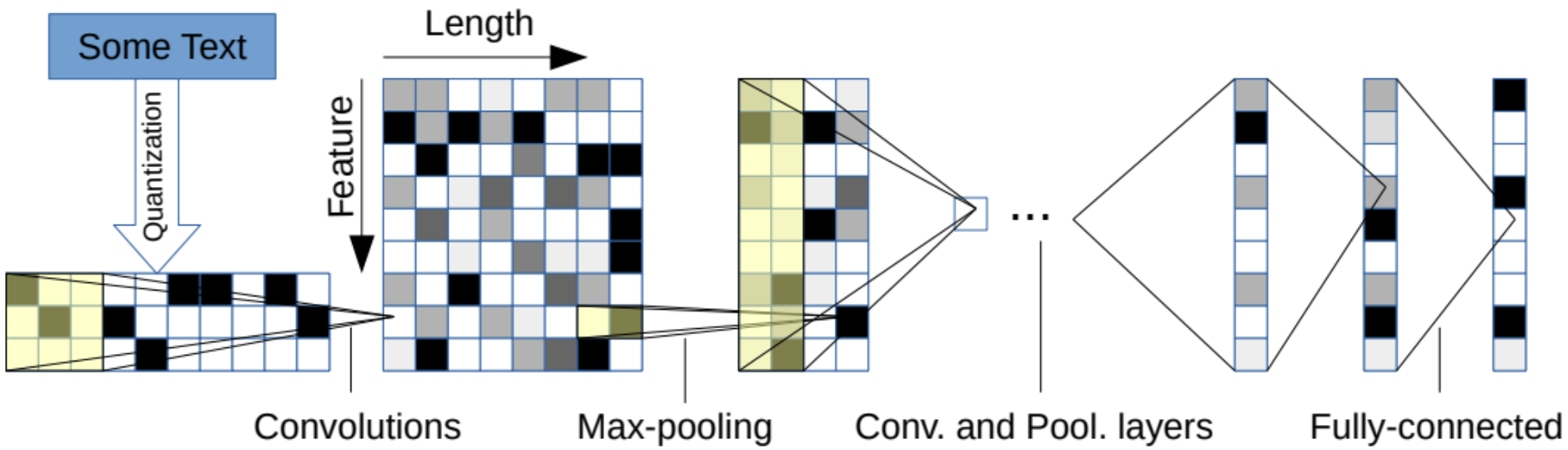
Dynamic Memory Network



Other models

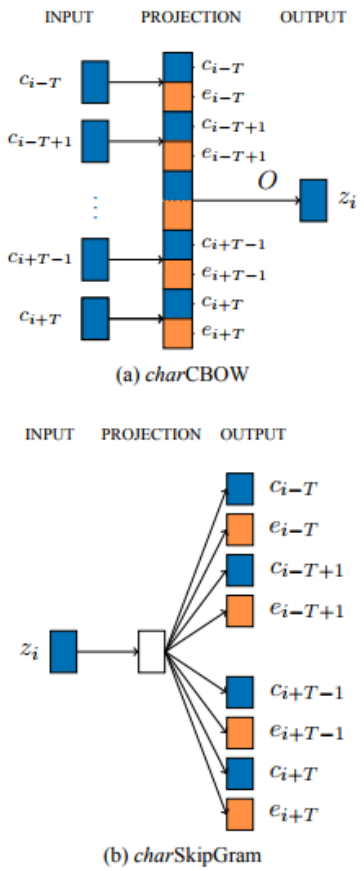
- Tree-LSTM
- Pointer networks
- Bi-Directional Block Self-Attention for Fast and Memory-Efficient Sequence Modeling (T. Shen et al., ICLR 2018)
- **Directional Self-Attention Network**
- Recurrent Entity Network

Char-CNN



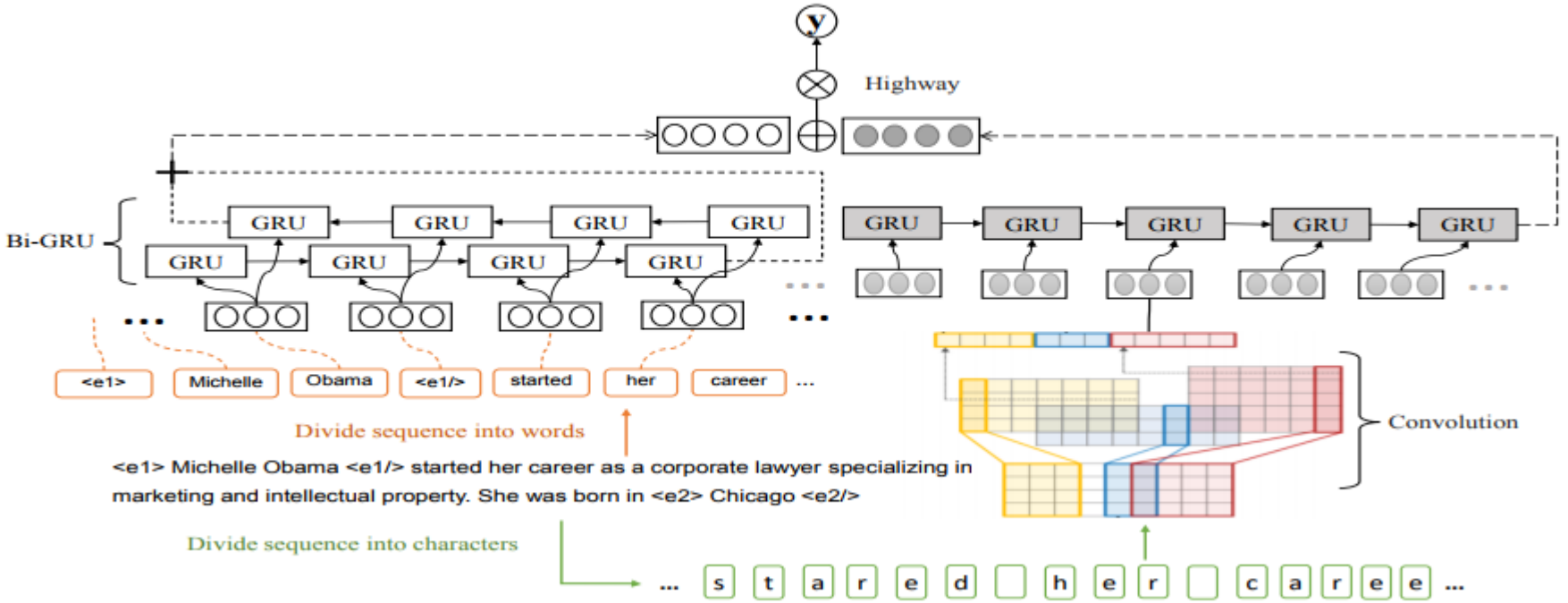
Zhang, Xiang, Junbo Zhao, and Yann LeCun. "Character-level convolutional networks for text classification." *Advances in neural information processing systems*. 2015.

Component-Enhanced



Component-Enhanced Yanran Li, Wenjie Li, Fei Sun, and Sujian Li. Component-Enhanced Chinese Character Embeddings. Proceedings of EMNLP, 201

Char-word Hybrids



Combining Word-Level and Character-Level Representations for Relation Classification of Informal Text

Long text/document classification

- **Hierarchical Attention Networks(HAN)**

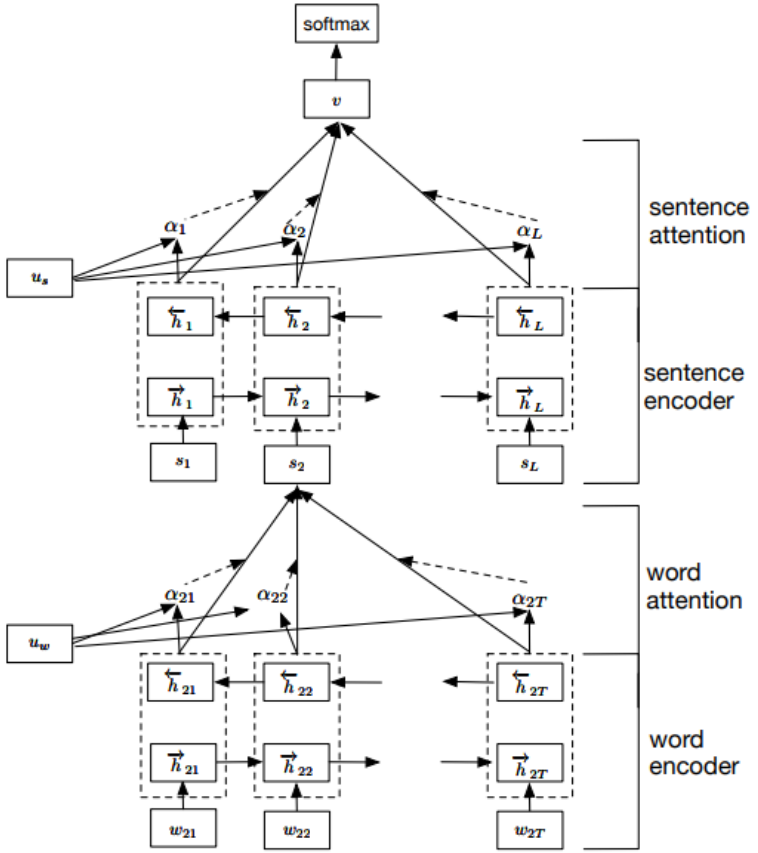
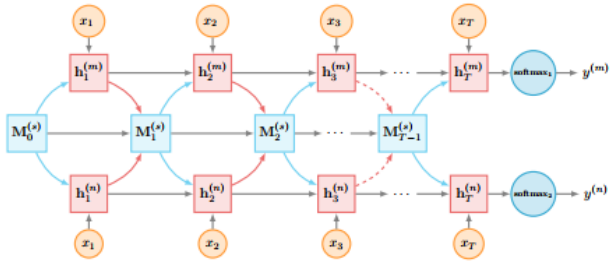
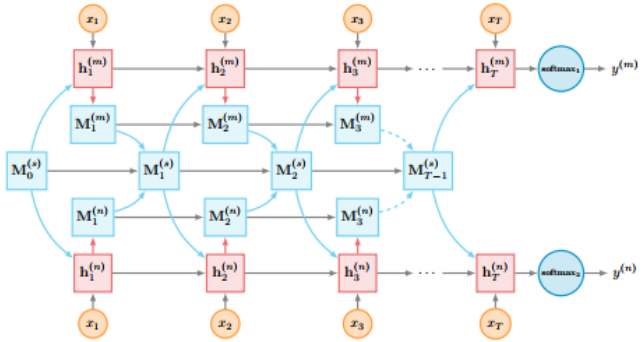


Figure 2: Hierarchical Attention Network.

Multi-task Learning



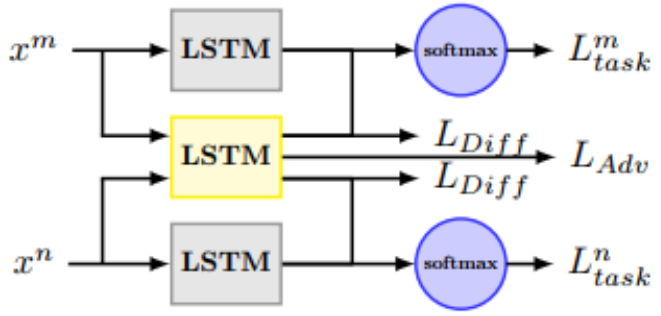
(a) Global Memory Architecture



(b) Local-Global Hybrid Memory Architecture

Pengfei Liu, **Xipeng Qiu**, Xuanjing Huang, Deep Multi-Task Learning with Shared Memory for Text Classification, In Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing **(EMNLP)**, 2016.

Adversarial Multi-task Learning

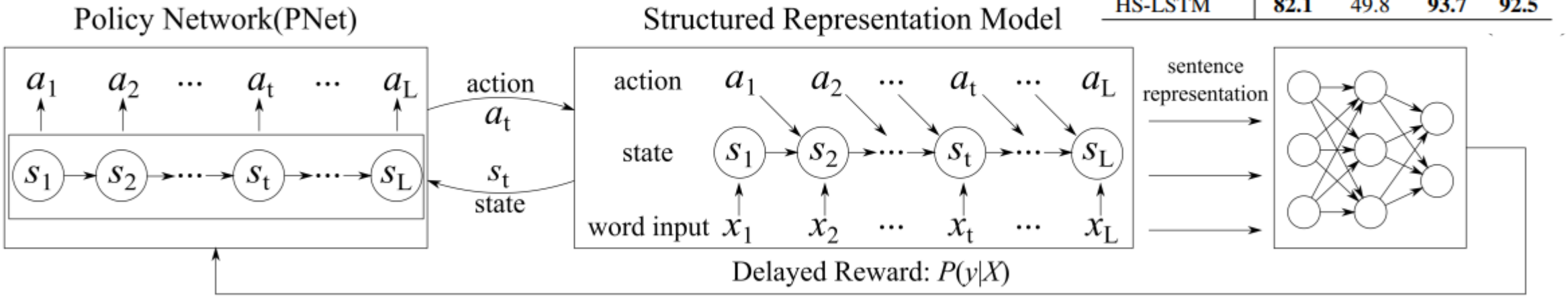


Pengfei Liu, **Xipeng Qiu**, Xuanjing Huang, Adversarial Multi-task Learning for Text Classification, In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics **(ACL)**, pp. 1-10, 2017.

RL for text classfication

- Learning Structured Representation for Text Classification via Reinforcement Learning AAAI 2018 minlieHuang

Models	MR	SST	Subj	AG
LSTM	77.4*	46.4*	92.2	90.9
biLSTM	79.7*	49.1*	92.8	91.6
CNN	81.5*	48.0*	93.4*	91.6
RAE	76.2*	47.8	92.8	90.3
Tree-LSTM	80.7*	**50.1**	93.2	91.8
Self-Attentive	80.1	47.2	92.5	91.1
ID-LSTM	81.6	50.0	93.5	92.2
HS-LSTM	**82.1**	49.8	**93.7**	**92.5**



Adversarial Training Methods For Semi-supervised Text Classification

Table 2: Test performance on the IMDB sentiment classification task. * indicates using pretrained embeddings of CNN and bidirectional LSTM.

Method	Test error rate
Baseline (without embedding normalization)	7.33%
Baseline	7.39%
Random perturbation with labeled examples	7.20%
Random perturbation with labeled and unlabeled examples	6.78%
Adversarial	6.21%
Virtual Adversarial	**5.91**%
Adversarial + Virtual Adversarial	6.09%
Virtual Adversarial (on bidirectional LSTM)	**5.91**%
Adversarial + Virtual Adversarial (on bidirectional LSTM)	6.02%
Full+Unlabeled+BoW (Maas et al., 2011)	11.11%
Transductive SVM (Johnson & Zhang, 2015b)	9.99%
NBSVM-bigrams (Wang & Manning, 2012)	8.78%
Paragraph Vectors (Le & Mikolov, 2014)	7.42%
SA-LSTM (Dai & Le, 2015)	7.24%
One-hot bi-LSTM* (Johnson & Zhang, 2016b)	5.94%

- ICLR 2017

To-do List

- Support more datasets, especially in Chinese
- Support more models
- Fine-tune the result.
- Installable Library with Python (Pip)